

Scilab Textbook Companion for  
Principles of Communication Systems  
by H. Taub and D. L. Schilling<sup>1</sup>

Created by  
Abhijeet Pasumarthu  
B.E in Electronics & Communication Engineering  
Electronics Engineering  
Birla Institute of Technology, Mesra  
College Teacher  
None  
Cross-Checked by  
Spandana

July 31, 2019

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

# Book Description

**Title:** Principles of Communication Systems

**Author:** H. Taub and D. L. Schilling

**Publisher:** Tmh

**Edition:** 3

**Year:** 2011

**ISBN:** 978-0-07-064811-1

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

**Exa** Example (Solved example)

**Eqn** Equation (Particular equation of the above book)

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

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

# Contents

List of Scilab Codes	4
1 Introduction of Signals and Spectra	5
2 Random Variables and Processes	12
3 Amplitude Modulation Systems	24
4 Angle Modulation	28
5 Pulse Modulation and Digital Transmission of Analog Signal	33
6 Digital Modulation and Transmission	41
7 Mathematical Representation of Noise	53
8 Noise in Amplitude Modulation System	56
9 Noise in Frequency Modulation Systems	61
10 Phase Locked Loops	66
11 Optimal Reception of Digital Signal	70

<b>12 Noise in Pulse Code Modulation and Delta Modulation Systems</b>	<b>79</b>
<b>13 Information Theory and Coding</b>	<b>86</b>
<b>14 Communication Systems and Component Noises</b>	<b>96</b>
<b>15 Spread Spectrum Modulation</b>	<b>100</b>

# List of Scilab Codes

Exa 1.1	Calculation of Energy . . . . .	5
Exa 1.2	Calculation of Power . . . . .	6
Exa 1.3	Time shifting . . . . .	6
Exa 1.4	Time shifting . . . . .	8
Exa 1.5	Crosscorrelation and Autocorrelation . . . . .	9
Exa 1.6	Autocorrelation . . . . .	10
Exa 2.1	Probability of two Events . . . . .	12
Exa 2.2	Bayes Theorem . . . . .	13
Exa 2.5	Optimum Receiver Algorithm . . . . .	14
Exa 2.6	Optimum Receiver Algorithm . . . . .	16
Exa 2.7	Optimum Receiver . . . . .	17
Exa 2.8	Probability Distribution Function . . . . .	18
Exa 2.10	Probability of Error . . . . .	19
Exa 2.11	Probability of Error . . . . .	20
Exa 2.13	Probability of Error . . . . .	22
Exa 3.4	Transmission Power Efficiency . . . . .	24
Exa 3.7	Calculation of L and C of Power Amplifier . . . . .	25
Exa 3.8	Calculation of Gain and Power radiated of Antenna . . . . .	26
Exa 4.2	Calculation of frequency parameters . . . . .	28
Exa 4.4	Calculation of Bandwidth . . . . .	30
Exa 5.2	Minimum sampling rate . . . . .	33
Exa 5.3	Calculation of Guard time . . . . .	34
Exa 5.5	Calculation of minimum number of Binary digits . . . . .	35
Exa 5.6	Companding . . . . .	36
Exa 5.9	LMS Algorithm . . . . .	37
Exa 5.11	SNR of a DM system . . . . .	38

Exa 6.1	Carrier phase variation . . . . .	41
Exa 6.2	DPSK and DEPSK . . . . .	45
Exa 6.4	Bandwidth and Noise susceptibility . . . . .	46
Exa 6.5	Duobinary decoding . . . . .	48
Exa 6.6	Roll off factor . . . . .	51
Exa 7.3	Noise Power . . . . .	53
Exa 7.4	SNR at output of equalizer . . . . .	54
Exa 8.1	SNR of SSB signal . . . . .	56
Exa 8.2	Signal strength and noise power density . . . . .	57
Exa 8.3	Minimum transmitter power . . . . .	58
Exa 8.4	SNR of a Square Demodulator . . . . .	59
Exa 9.1	SNR of an FM Limiter Discriminator Demodulator . . . . .	61
Exa 9.2	FM Limiter Discriminator Demodulator . . . . .	62
Exa 9.3	RC Filter Preemphasis Deemphasis . . . . .	63
Exa 9.6	SNR at Input . . . . .	64
Exa 10.3	SNR of Phase Discriminator . . . . .	66
Exa 10.5	Channel Spacing . . . . .	67
Exa 10.7	Phase Locked Loops . . . . .	68
Exa 11.3	Optimal threshold and probability of error . . . . .	70
Exa 11.4	Decision threshold . . . . .	72
Exa 11.5	Probability of error of optimum filter . . . . .	73
Exa 11.6	Error probability of BPSK signal . . . . .	74
Exa 11.7	Error probability of coherent FSK signal . . . . .	76
Exa 11.8	Error probability in Optimal Reception . . . . .	77
Exa 11.9	Probability of error of QPR System . . . . .	78
Exa 12.2	SNR Optimal receiver . . . . .	79
Exa 12.3	SNR of Optimal reciver . . . . .	81
Exa 12.4	Output SNR in DM including Thermal Noise . . . . .	84
Exa 13.1	Information rate of source . . . . .	86
Exa 13.4	Channel Capacity . . . . .	87
Exa 13.8	Probability of error . . . . .	88
Exa 13.11	Turbo code . . . . .	89
Exa 13.12	Turbo code . . . . .	90
Exa 13.14	GO back N algorithm . . . . .	92
Exa 13.15	Power of a Transmitter . . . . .	93
Exa 13.16	Probability of error for Trellis decoded modulation . . . . .	94

Exa 14.1	Thermal noise voltage . . . . .	96
Exa 14.2	Output Noise power . . . . .	97
Exa 14.3	Transmitted power of antenna . . . . .	98
Exa 15.1	Jamming . . . . .	100
Exa 15.2	Ranging using DS spread spectrum . . . . .	101
Exa 15.3	Sequence length . . . . .	102



# Chapter 1

## Introduction of Signals and Spectra

Scilab code Exa 1.1 Calculation of Energy

```
1  clc;  
2  //page 12  
3  //problem 1.1  
4  
5  //Given signal  $u = 2*\exp(-3*t)$   
6  
7  //Since the function integral does not accept %inf  
   as limit we need to use approximation by changing  
   variables.  
8  
9  //First the signal is to be expressed in terms of 'x'  
   ' .  
10  
11 function y=Signal(x);  
12 y=2*exp(-3*x);  
13 endfunction;  
14  
15 //We then substitute  $x = \tan(z)$ , and then express  
   the given signal wrt 'z' and not 'x'.
```

```

16
17 function y=Gmodified(z);
18 x=tan(z);
19 y=(Signal(x))^2/(cos(z))^2;
20 endfunction;
21
22 E = intg(0,atan(10),Gmodified)
23
24 disp(E,'The energy of this signal is ');

```

---

#### Scilab code Exa 1.2 Calculation of Power

```

1 clc;
2 //page 12
3 //problem 1.2
4
5 //Given signal u = 2*sin(0.5*%pi*t)
6
7 //Since u is periodic, averaging over -infinity to +
  infinity will give the same result as averaging
  over -2 to 2, where 4 is the time period.
8
9 t0 = -2;
10 t1 = 2;
11 E = integrate('(2*sin(0.5*%pi*t))^2','t',t0,t1)/4;
12
13 disp(E,'The power of the signal is ');

```

---

#### Scilab code Exa 1.3 Time shifting

```

1 clc;
2 //page 18
3 //problem 1.3

```

```

4
5 //u1(T) vs T
6 T = [-5:0.0082:5];
7 u1(T<=0) = 0;
8 u1(T>0) = 1;
9 xlabel('T');
10 ylabel('u(T)')
11
12 subplot(131);
13 plot2d(T,u1);
14
15 //u2(T-t) vs T
16 //Shifting the given signal by t units to the right,
    we get
17 //Let us assume the amount of time to be shifted is 3
    units
18 t = 3;
19
20 T = [-5:0.0082:5];
21 u2(T<=t) = 0;
22 u2(T>t) = 1;
23 xlabel('T');
24 ylabel('u(T - t)')
25
26 subplot(132);
27 plot2d(T,u2);
28
29 //u(t - T) = u(-(T - t))
30
31 T = [-5:0.0082:5];
32 u3(T>=t) = 0;
33 u3(T<t) = 1;
34 xlabel('T');
35 ylabel('u(t - T)')
36
37 subplot(133);
38 plot2d(T,u3);

```

---

### Scilab code Exa 1.4 Time shifting

```
1  clc;
2  //page 18
3  //problem 1.4
4
5  //u1(t)
6  t = [-5:0.0082:5];
7  u1(t<=0) = 0;
8  u1(t>0) = 1;
9
10 xlabel('t');
11 ylabel('u(t)')
12
13 subplot(131);
14 plot2d(t,u1);
15
16 //u2(t-T)
17 //Shifting the given signal by t units to the right,
    we get
18 //Let us assume the amount of time to be shifted is 3
    units
19 T = 3;
20
21 t = [-5:0.0082:5];
22 u2(t<=T) = 0;
23 u2(t>T) = 1;
24
25 xlabel('t');
26 ylabel('u(t-T)')
27
28 subplot(132);
29 plot2d(t,u2);
30
```

```

31
32 //u(t) - u(t - T)
33
34 t = [-5:0.0082:5];
35 u3 = u1 - u2;
36
37 xlabel('t');
38 ylabel('u(t) - u(t-T)')
39
40 subplot(133);
41 plot2d(t,u3);

```

---

#### Scilab code Exa 1.5 Crosscorrelation and Autocorrelation

```

1  clc;
2  //page 18
3  //problem 1.5
4
5  //V1(t) = u(t) - u(t - 5)
6  t = [-5:0.1:5];
7  V1(t<=0) = 0;
8  V1(t>0) = 1;
9
10 xlabel('t');
11 ylabel('V1(t)')
12 subplot(121);
13 plot2d(t,V1);
14
15
16 //V2(t) = 2*t*(u(t) - u(t - 3))
17 t = [0:0.1:3];
18 V2 = 2*t;
19
20 xlabel('t');
21 ylabel('V2(t)')

```

```

22 subplot(122);
23 plot2d(t,V2);
24
25 //Autocorrelation R12(0) = R
26
27 R = integrate('2*t','t',0,3);
28
29 E1 = integrate('1','t',0,5);
30
31 //In the textbook, E2 has been computed as 18
    instead of 36
32 E2 = integrate('4*t^2','t',0,3);
33
34 c = R/(E1*E2)^0.5;
35
36 disp(R,'The correlation term R12(0) is ');
37 disp(E1,'The autocorrelation term R1(0) is ');
38 disp(E2,'The autocorrelation term R2(0) is ');

```

---

#### Scilab code Exa 1.6 Autocorrelation

```

1  clc;
2  //page 19
3  //problem 1.6
4
5  //V1(t) = u(t) - u(t - 5)
6  t = [-5:0.1:5];
7  V1(t<=0) = 0;
8  V1(t>0) = 1;
9
10 xlabel('t');
11 ylabel('V1(t)');
12 subplot(121);
13 plot2d(t,V1);
14

```

```

15
16 //V2(t) = 2*t*(u(t) - u(t - 3))
17 t = [0:0.1:3];
18 V2 = 2*t;
19
20 xlabel('t');
21 ylabel('V2(t)')
22 subplot(122);
23 plot2d(t,V2);
24
25 //Autocorrelation R12(1) = Ra
26 //The range is t = 0 to 2, as signal V2(t) has been
    shifted left by one unit, V2(t-1)
27
28 Ra = integrate('2*(t+1)', 't', 0, 2);
29
30 disp(Ra, 'The correlation term R12(1) is ');
31
32 //Autocorrelation R12(-1) = Rb
33 //The range is t = 1 to 4, as signal V2(t) has been
    shifted right by one unit, V2(t+1)
34
35 Rb = integrate('2*(t-1)', 't', 1, 4);
36
37 disp(Rb, 'The correlation term R12(-1) is ');

```

---

## Chapter 2

# Random Variables and Processes

Scilab code Exa 2.1 Probability of two Events

```
1  clc ;
2  //page 85
3  //problem 2.1
4
5  //A & B are two events occurred in sample space S,
   where P(A) & P(B) are their corresponding
   probability
6  P_S=1
7
8  //Given A&B are not mutually exclusive events ,
9  //Probability of A is 0.2 = P_A
10 //Probability of B is 0.4 = P_B
11 //Probability of either A or B is 0.5 = P_AUB
12 P_A = 0.2
13 P_B = 0.4
14 P_AUB = 0.5
15
16 //Probability of both of A&B jointly occur is
   P_AinterB = P_A+P_B-P_AUB where inter is
```



```

        intersection
17 P_AinterB = P_A+P_B-P_AUB
18
19 disp('Probability of both of A&B jointly occur is '+
        string(P_AinterB))
20
21 //Probability of none of AorB are occur is P_NOAB =
        Total occurence(P_S) - Probability of either AorB
        (P_AUB)
22 P_NOAB = P_S-P_AUB
23
24 disp('Probability of none of AorB are occur is '+
        string(P_NOAB))

```

---

#### Scilab code Exa 2.2 Bayes Theorem

```

1  clc;
2  //page 86
3  //problem 2.2
4
5  //Probability that A will occur if B has already
        occurred(P_AB) = ratio of Probability of joint
        occurence of A&B (P_A_B) & Probability of B(P_B)
6  //P_A_B(robability of joint occurence) = Probability
        that A&B both occur(P_AinterB)
7
8  //From given values P_AinterB = 0.1 implies P_A_B =
        0.1 & P_B = 0.4
9  P_AinterB = 0.1
10 P_A_B = P_AinterB
11 P_B = 0.4
12
13 P_AB = P_A_B/P_B
14
15 //Similarly

```

```

16 //Probability that B will occur if A has already
    occurred(P_BA) = ratio of Probability of joint
    occurrence of A&B (P_A_B) & Probability of B(P_A)
17
18 //From given values P_A = 0.2
19 P_A = 0.2
20
21 P_BA = P_A_B/P_A
22
23 //Bayes theorem says that P_AB = (P_A/P_B)*P_BA
24 //After Calculating LHS & RHS if both are equal then
    bayes theorem is satisfying
25
26 //Calculating LHS
27 LHS = P_AB
28
29 //Calculating RHS
30 RHS = (P_A/P_B)*P_BA
31
32 disp('P(A/B) = '+string(P_AB) );
33
34 if LHS == RHS then
35     disp('LHS = RHS, Hence Bayes theorem is verified
        ');
36 end

```

---

#### Scilab code Exa 2.5 Optimum Receiver Algorithm

```

1  clc;
2  //page 95
3  //problem 2.5
4
5  //Given, The probability that m0 is sent is 0.7 =
    P_m0
6  P_m0 = 0.7

```

```

7
8 //The probability that m0 is sent is 0.3 = P_m1
9 P_m1 = 0.3
10
11 //The probability that r0 is received given that m0
    is sent is 0.9 = P_r0m0 where r is voltage & m is
    message
12 P_r0m0 = 0.9
13
14 //the probability that r1 is received given that m0
    is sent is 0.1 = P_r1m0 where r is voltage & m is
    message
15 P_r1m0 = 0.1
16
17 //The probability that r1 is received given that m1
    is sent is 0.6 = P_r1m1
18 P_r1m1 = 0.6
19
20 //the probability that r0 is received given that m1
    is sent is 0.4 = P_r0m1 where r is voltage & m is
    message
21 P_r0m1 = 0.4
22
23 //With the given values check eqations P_r0m0*P_m0(
    P00) > P_r0m1*P_m1(P01)
24 P00 = P_r0m0*P_m0
25 P01 = P_r0m1*P_m1
26
27 if P00>P01 then
28     disp('as P(r0|m0)*P(m0) > P(r0|m1)*P(m1) is
        valid , we whould select m0 whenever r0 is
        received')
29 end
30
31 //With the given values check eqations P_r1m1*P_m1(
    P11) > P_r1m0*P_m0(P10)
32 P11 = P_r1m1*P_m1
33 P10 = P_r1m0*P_m0

```

```

34
35 if P11>P10 then
36     disp('as  $P(r1|m1)*P(m1) > P(r1|m0)*P(m0)$  is
        valid, we should select m1 whenever r1 is
        received')
37 end

```

---

### Scilab code Exa 2.6 Optimum Receiver Algorithm

```

1 clc;
2 //page 96
3 //problem 2.6
4
5 //Given, the probability that r0 is received given
   that m0 is sent is  $0.9 = P_{r0m0}$  where r is
   voltage & m is message
6 P_r0m0 = 0.9
7
8 //The probability that m0 is sent is  $0.7 = P_{m0}$ 
9 P_m0 = 0.7
10
11 //The probability that r1 is received given that m1
   is sent is  $0.6 = P_{r1m1}$ 
12 P_r1m1 = 0.6
13
14 //The probability that m0 is sent is  $0.3 = P_{m1}$ 
15 P_m1 = 0.3
16
17 //The probability that the transmitted signal is
   correctly read at receiver is  $P(c)(P_c) =$  the
   probability that m0 was sent when r0 was read(
    $P_{r0m0}*P_{m0}$ ) + the probability that m1 was sent
   when r1 was read( $P_{r1m1}*P_{m1}$ )
18
19 P_c = P_r0m0*P_m0+P_r1m1*P_m1

```

```

20
21 //P(e)(P_e) = 1-P(c)
22 P_e = 1-P_c
23
24 disp('P(e) = '+string(P_e))
25 disp('P(c) = '+string(P_c))

```

---

### Scilab code Exa 2.7 Optimum Receiver

```

1  clc;
2  //page 96
3  //problem 2.7
4
5  //Here P(ra|mb) is denoted as P_ramb where a is
   0,1,2 & b is 0,1
6  //P(X) is denoted as P_X where X is m0, m1, C & E
7
8  //From given values P_m0 = 0.6, P_m1 = 0.4, P_r0m0
   =0.6, P_r1m1 = 0.7, P_r0m1 = 0, P_r1m0 = 0.2,
   P_r2m0 = 0.2 & P_r2m1 = 0.3
9  P_m0 = 0.6
10 P_m1 = 0.4
11 P_r0m0 =0.6
12 P_r1m1 = 0.7
13 P_r0m1 = 0
14 P_r1m0 = 0.2
15 P_r2m0 = 0.2
16 P_r2m1 = 0.3
17
18 //(a)
19 //Comaparing P(r0|m0)*P(m0) & P(r0|m1)*P(m1) gives
   result
20 LHS = P_r0m0*P_m0
21 RHS = P_r0m1*P_m1
22

```

```

23 disp('As  $P(r_0|m_0)*P(m_0)$  [ '+string(LHS)+' ] >  $P(r_0|m_1)*$ 
      P(m1) [ '+string(RHS)+' ] ')
24 disp('we select m0 whenever r0 is received')
25
26 //Similarly compare  $P(r_1|m_1)*P(m_1)$  &  $P(r_1|m_0)*P(m_0)$ 
27 LHS = P_r1m1*P_m1
28 RHS = P_r1m0*P_m0
29
30 disp('As  $P(r_1|m_1)*P(m_1)$  [ '+string(LHS)+' ] >  $P(r_1|m_0)*$ 
      P(m0) [ '+string(RHS)+' ] ')
31 disp('we select m1 whenever r1 is received')
32
33 //compare  $P(r_2|m_0)*P(m_0)$  &  $P(r_2|m_1)*P(m_1)$ 
34 LHS = P_r2m0*P_m0
35 RHS = P_r2m1*P_m1
36
37 disp('As  $P(r_2|m_0)*P(m_0)$  [ '+string(LHS)+' ] =  $P(r_2|m_1)*$ 
      P(m1) [ '+string(RHS)+' ] ')
38 disp('We can accordingly make either assignment and
      we arbitrarily associate r2 with m0')
39
40 //(b)
41 //The probability of being correct is  $P(C) = P(r_0|m_0)$ 
       $*P(m_0)+P(r_1|m_1)*P(m_1)+P(r_2|m_0)*P(m_0)$ 
42 P_C = P_r0m0*P_m0+P_r1m1*P_m1+P_r2m0*P_m0
43
44 //The probability of error is  $P(E) = 1-P(C)$ 
45 P_E = 1 - P_C;
46
47 disp('Probability of being correct is  $P(C) =$ ' +
      string(P_C)')
48 disp('Probability of error is  $P(E) =$ ' +string(P_E)')

```

---

Scilab code Exa 2.8 Probability Distribution Function

```

1  clc;
2  //page 99
3  //problem 2.8
4
5  //Given, probability density function of X is fX_x
   where fX_x = a*e^(-0.2*x) for x greater than &
   equal to 0 & = 0 elsewhere
6
7  //a = fX_x/(a*e^(-0.2*x))
8  //from definition integration of fX_x with limits -
   infinity to +infinity is 1
9  //As per given fX_x, integration of a*e^(-0.2*x)
   with limits 0 & +infinity and obtained value be
   P
10 //a = 1/p
11
12 P = integrate('%e^(-0.2*x)', 'x', 0, 100)
13 a = 1/P
14
15 disp('a = '+string(a))

```

---

#### Scilab code Exa 2.10 Probability of Error

```

1  clc;
2  //page 105
3  //problem 2.10
4
5  //We know that, Probabilty of error(P_error) for the
   signal correpted by Gaussian channel variance
   sigma^2 where signal having voltage levels as 0&V
   is (1/2)*erfc(V/(2*sqrt(2)*sigma))
6
7  //P_error for V = 4 & sigma^2 =2
8  V = 4
9  sigma = sqrt(2)

```

```

10 P_error = (1/2)*erfc(V/(2*sqrt(2)*sigma))
11
12 disp('Probabilty of error for V = 4 & sigma^2 =2 is
      '+string(P_error))
13
14 //P_error for V = 2 & sigma^2 =2
15 V = 2
16 sigma = sqrt(2)
17 P_error = (1/2)*erfc(V/(2*sqrt(2)*sigma))
18
19 disp('Probabilty of error for V = 2 & sigma^2 =2 is
      '+string(P_error))
20
21 //P_error for V = 4 & sigma^2 =4
22 V = 4
23 sigma = sqrt(4)
24 P_error = (1/2)*erfc(V/(2*sqrt(2)*sigma))
25
26 disp('Probabilty of error for V = 4 & sigma^2 =4 is
      '+string(P_error))
27
28 //P_error for V = 8 & sigma^2 =2
29 V = 8
30 sigma = sqrt(2)
31 P_error = (1/2)*erfc(V/(2*sqrt(2)*sigma))
32
33 disp('Probabilty of error for V = 8 & sigma^2 =2 is
      '+string(P_error))

```

---

### Scilab code Exa 2.11 Probability of Error

```

1 clc;
2 //page 106
3 //problem 2.11
4

```



```

5 //(a)
6 //out of n attempts the probability of message
  reaching correctly for k times is given by
  binomial distribution  $pX(k) = nCk*(q^k)*(1-q)^{(n-k)}$ 
  where q is probability of correctly reaching
7
8 //Here n = 10, k = 1, q = 0.001
9 n = 10
10 k = 1
11 q = 0.001
12
13 //pX(k) is denoted as p_X_1
14 //10C1 =10
15 p_X_1 = 10*(q^k)*(1-q)^(n-k)
16
17 disp('The probability that out of 10 transmissions 9
  are correct and 1 is incorrect is '+string(p_X_1
  ))
18
19 //probability that more than two erroneous out of
  100 transmissions (p_100_2) = 1-probability of
  less than or equal to two error in transmission
20 //p_100_2 = 1-pX(0)-pX(1)-pX(2)
21 //p_100_2 =1-100C0*((0.001)^0)*((1-0.001)^100)-100C1
  *((0.001)^1)*((1-0.001)^99)-100C0*((0.001)^2)
  *((1-0.001)^98)
22
23 //Since, calculation of above is cumbersome we may
  use Poisson ditribution to approximate above
24 //Poisson distribution =  $pX(k) = (alfa^k)*(e^{-alfa})/k!$ ,
  where alfa = n*q
25
26 //Here n = 100 & q = 0.001
27 n = 100
28 q = 0.001
29
30 alfa = n*q
31

```

```

32 p_100_2 = 1-(alfa^0)*(%e^-0.1)/factorial(0)-(alfa^1)
    *(%e^-0.1)/factorial(1)-(alfa^2)*(%e^-0.1)/
    factorial(2)
33
34 disp('probability that more than two erroneous out
    of 100 transmissions is '+string(p_100_2))
35
36 //(c)
37 //from(b), required probability i.e probability of
    more than one are erroneous out of 100
    transmission(p_100_1) is
38 p_100_1 = 1-(alfa^0)*(%e^-0.1)/factorial(0)-(alfa^1)
    *(%e^-0.1)/factorial(1)
39
40 disp('probability that more than one erroneous out
    of 100 transmissions is '+string(p_100_1))

```

---

### Scilab code Exa 2.13 Probability of Error

```

1  clc;
2  //page 115
3  //problem 2.13
4
5  //Given, Error probability is  $10^{-4} = P_e$ , no of
    ecperiments conducted =  $N = 4*10^5$  & estimated
    probability of error p does not differ from  $P_e$ 
    by more than 50%
6  P_e = 10^-4
7  N = 4*10^5
8
9  //Tchebycheff's inequality is  $P(|p-P_e| \geq E) \leq P_e / (N * E^2)$ 
10 //From given values we can find that  $E = 50*10^{-4}$ 
11 E = 50*10^-4
12

```

```

13 //Here R.H.S of Tchebycheff's inequality is denoted
    as Tc_RHS
14 Tc_RHS = P_e/(N*E^2)
15
16 //Tc_RHS in percentage is Tc_RHSper
17 Tc_RHSper = Tc_RHS/100
18
19 //disp(Tc_RHSper,Tc_RHS,'or P(|p-10^-4|>=0.5*10^-2)
    <=','Tc_RHS,'The probability of estimated
    probability of error p does not differ from P_e
    by more than 50% is less than equal to')
20
21 //given solution has been computed wrong, obtaines
    solution is 10^-7
22 disp('The probability of estimated probability of
    error p does not differ from P_e by more than 50%
    is less than equal to '+string(Tc_RHS)+'or P(|p
    -10^-4|>=0.5*10^-2)<=','+string(Tc_RHS)+' = '+
    string(Tc_RHSper)+'%')

```

---

# Chapter 3

## Amplitude Modulation Systems

Scilab code Exa 3.4 Transmission Power Efficiency

```
1  clc;  
2  //page 163  
3  //problem 3.4  
4  
5  //Transmission power efficiency  $n = \{(m^2) / [2+(m^2)]\} * 100\%$  where m is modulated index  
6  
7  //Given modulated indices are m1 = 0.25, m2 = 0.5 &  
   m3 = 0.75  
8  
9  //Transmission power efficiencies are n1, n2 & n3  
   respectively for m1, m2 & m3  
10 n1 =  $\{(0.25^2) / [2+(0.25^2)]\} * 100$   
11 n2 =  $\{(0.5^2) / [2+(0.5^2)]\} * 100$   
12 n3 =  $\{(0.75^2) / [2+(0.75^2)]\} * 100$   
13  
14 disp('Transmission power efficiency for modulated  
   index 0.25 is '+string(n1)+ '%')  
15 disp('Transmission power efficiency for modulated  
   index 0.5 is '+string(n2)+ '%')  
16 disp('Transmission power efficiency for modulated
```

```
index 0.75 is '+string(n3)+ '%')
```

---

### Scilab code Exa 3.7 Calculation of L and C of Power Amplifier

```
1  clc;
2  //page 185
3  //problem 3.7
4
5  //Given input inmedance of matching networkis R1 =
   10 ohm & output impedance of matching networ is
   R2 = 50 ohm & carrier frequency is fc = 500 KHz
6  R1 = 10
7  R2 = 50
8  fc = 500000
9
10 //Wc = 2*pi*fc
11 Wc = 2*%pi*fc
12
13 //AS  $R1 = R2*(X2^2) / [(R2^2)+(X2^2)]$ ,  $X2 = 25\text{ohm}$ 
14 X2 = 25
15
16 //AS  $X1 = (R2^2)*X2 / [(R2^2)+(X2^2)]$  &  $R1>R2$ ,  $X1 =$ 
   -20ohm
17 X1 = -20
18
19 //  $|X1| = |jwL| = wL = 20$  &  $|X2| = |1/jwC| = 1/wC =$ 
   25, so  $|X1*X2| = L/C = 500$  denotes as LC_div
20 LC_div = 500
21
22 //  $Wc^2 = 1/(L*C)$ . LC is denoted as LC_prod
23 LC_prod = 1/(Wc^2)
24
25 //In the textbook the calculated  $LC = 10^{-3}$ , in
   reality the value of  $LC = 1.013D-13$ 
26
```

```

27 L = sqrt(LC_div*LC_prod)
28
29 //In the textbook the calculated  $L^2 = 50 \cdot 10^{-14}$ , in
    reality the value of  $L^2 = 5.066D-11$ 
30
31 C = L/500
32
33 //In the textbook the calculated  $C = 1.4 \cdot 10^{-9}$ , in
    reality the value of  $C = 1.424D-08$ 
34
35 disp('Inductance '+string(L)+' H')
36 disp('Capacitance '+string(C)+' F')

```

---

**Scilab code Exa 3.8** Calculation of Gain and Power radiated of Antenna

```

1  clc;
2  //page 185
3  //problem 3.8
4
5  //Given ohmic loss resistance is  $R_o = 12$  Ohm,
6  Ro = 12
7
8  //radiation resistance is  $R_r = 48$  Ohm,
9  Rr = 48
10
11 //directivity is  $D = 2$ 
12 D = 2
13
14 //Input current =  $0.1 \cdot \cos[2 \cdot \pi \cdot (10^6) \cdot t]$ , Amplitude
    of input current is  $A = 0.1$  Amp
15 A = 0.1
16
17 //Equivalent resistance =  $R_e = R_o + R_r$ 
18 Re = Ro+Rr
19

```

```
20 //Total power used in antenna = Pin = (A^2)*Re/2
21 Pin = (A^2)*Re/2
22
23 //Power used in radiation = Prad = (A^2)*Rr/2
24 Prad = (A^2)*Rr/2
25
26 //Efficiency of the antenna = n = Prad/Pin
27 n = Prad/Pin
28
29 //Gain of antenna = Ga = efficiency*directivity
30 Ga = n*D
31
32 disp('Total power used in antenna '+string(Pin)+'
      Watt')
33 disp('Power used in radiation '+string(Prad)+' Watt'
      )
34 disp('Efficiency of the antenna '+string(n))
35 disp('Gain of antenna '+string(Ga))
```

---

# Chapter 4

## Angle Modulation

Scilab code Exa 4.2 Calculation of frequency parameters

```
1  clc ;
2  //page 199
3  //problem 4.2
4
5  //Given angle modulated signal is  $x = 3 \cos [2 \pi$ 
    $*(10^6)*t + 2 \sin (2 \pi * 10^3 * t)]$ 
6
7  //So, phase of the angle modulates signal is  $Q = 2 \pi$ 
    $*(10^6 * t) + 2 \sin (2 \pi * (10^3) * t)$ 
8
9  //Instantaneous frequency =  $dQ/dt = 2 \pi * (10^6) + 4 \pi$ 
    $*(10^3) * \sin (2 \pi * (10^3) * t)$ 
10
11 //For Instantaneous frequency at 0.25ms,
   Substituting  $t = 0.25 \text{ms}$  in Instantaneous
   frequency
12 //Instantaneous frequency is expressed as  $f_{1\_rad}$  for
   frequency in radians per second
13  $f_{1\_rad} = 2 * \% \pi * (10^6) + 4 * \% \pi * (10^3) * \sin (2 * \% \pi * (10^3)$ 
    $* 0.00025)$ 
14
```



```

15 //Instantaneous frequency is expressed as f1_hz for
    frequency in hertz
16 f1_hz = f1_rad/(2*%pi)
17
18 disp('the Instantaneous frequency at time t=0.25ms
    is '+string(f1_rad)+' rad/sec = '+string(f1_hz)+'
    Hz')
19
20 //For Instantaneous frequency at 0.25ms,
    Substituting t = 0.5ms in Instantaneous frequency
21 //Instantaneous frequency is expressed as f2rad for
    frequency in radians per second
22 f2_rad = 2*%pi*(10^6)+ 4*%pi*(10^3)*sin(2*%pi*(10^3)
    *0.0005)
23
24 //Instantaneous frequency is expressed as f2hz for
    frequency in hertz
25 f2_hz = f2_rad/(2*%pi)
26
27 disp('the Instantaneous frequency at time t=0.5ms is
    '+string(f2_rad)+' rad/sec = '+string(f2_hz)+'
    Hz')
28
29 //Maximum phase deviation = max[2*sin(2*pi*(10^3)*t)
    ] = 2*1
30 maxDp = 2;
31
32 disp('Maximum phase deviation is '+string(maxDp)+'
    rad')
33
34 //Maximum frequency deviation = max[4*pi*(10^3)*sin
    (2*pi*(10^3)*t)] = 4*pi*(10^3)*1
35 maxDf = 4*%pi*(10^3)*1;
36
37 disp('Maximum frequency deviation is '+string(maxDf)
    +' Hz')
38 //disp('in rad',maxDf,'Maximum frequency deviation
    is ')

```

39

```
40 //In the textbook the calculated value of max
    frequency deviation is = 2000 Hz, in reality the
    value = 12566.371 Hz
```

---

#### Scilab code Exa 4.4 Calculation of Bandwidth

```
1  clc;
2  //page 208
3  //problem 4.4
4
5  //Given modulating signal  $m(t) = 2*\sin(2*\pi*(10^3)*t$ 
    ), B for phase modulation  $B_p = 10$  & for frequency
    modulation  $B_f = 10$ 
6  Bp = 10
7  Bf = 10
8
9  //So Amplitude of modulating signal is  $A_m=2$  metres
10 Am = 2
11
12 //Frequency of modulating signal is  $f_m = 1000$  hertz
13 fm=1000
14
15 //Bandwidth =  $2*(1+B)*f_m$ 
16
17 //Bandwidth for phase modulation with modulating
    signal  $m(t)$  is  $bw_{pm} = 2*(1+B_p)*f_m$ 
18 bw_pm = 2*(1+10)*1000
19
20 //Bandwidth for frequency modulation with modulating
    signal  $m(t)$  is  $bw_{fm} = 2*(1+B_f)*f_m$ 
21 bw_fm = 2*(1+10)*1000
22
23 disp('Bandwidth for phase modulation '+string(bw_pm)
    +' Hz')
```

```

24 disp('Bandwidth for frequency modulation '+string(
    bw_fm)+' Hz')
25
26 //Bandwidth for phase & frequency modulation if
    frequency of modulating signal is doubled i.e fm
    = 2000 hertz
27
28 //Bp & Bf after frequency of modulating signal is
    doubled
29
30 //Bp = kp*Am, observing the equation as there is no
    change in amplitude Bp = 10
31 Bp = 10
32
33 //Bf = kf*Am/fm, observing the equation as there is
    change in frequency Bf = 10/2 = 5
34 Bf = 5
35
36 //Bandwidth for phase modulation if frequency of
    modulating signal is doubled is bw_double_pm =
    2*(1+Bp)*fm
37 bw_double_pm = 2*(1+10)*2000
38
39 //Bandwidth for frequency modulation if frequency of
    modulating signal is doubled is bw_double_fm =
    2*(1+Bf)*fm
40 bw_double_fm = 2*(1+5)*2000
41
42 disp('Bandwidth for phase modulation for doubled
    frequency '+string(bw_double_pm)+' Hz')
43 disp('bandwidth for frequency modulation for doubled
    frequency '+string(bw_double_fm)+' Hz')
44
45 //Bandwidth for phase & frequency modulation if
    amplitude of modulating signal is halfed i.e Am =
    1 metre
46
47 //Bp & Bf after amplitude of modulating signal is

```

```

    halfed
48
49 //Bp = kp*Am, observing the equation as there is
    change in amplitude Bp = 10/2 = 5
50 Bp = 5
51
52 //Bf = kf*Am/fm, observing the equation as there is
    change in amplitude Bf = 5/2 = 2.5
53 Bf = 2.5
54
55 //Bandwidth for phase modulation if frequency of
    modulating signal is doubled is bw_halfed_pm =
    2*(1+Bp)*fm
56 bw_halfed_pm = 2*(1+5)*2000
57
58 //Bandwidth for frequency modulation if frequency of
    modulating signal is doubled is bw_halfed_fm =
    2*(1+Bf)*fm
59 bw_halfed_fm = 2*(1+2.5)*2000
60
61 disp('Bandwidth for phase modulation for halfed
    amplitude '+string(bw_halfed_pm)+' Hz')
62 disp('Bandwidth for frequency modulation for halfed
    amplitude '+string(bw_halfed_fm)+' Hz')

```

---

# Chapter 5

## Pulse Modulation and Digital Transmission of Analog Signal

Scilab code Exa 5.2 Minimum sampling rate

```
1  clc;
2  //page 247
3  //problem 5.2
4
5  //Highest frequency(fH) = 10000/2 = 5000 Hz
6  fH = 5000
7
8  //Lowest frequency(fL) = 6000/2 = 3000 Hz
9  fL = 3000
10
11 //Minimum sampling frequency from low pass
    consideration(SLOW) = 2*fH
12 S_LOW = 2*fH
13
14 disp('Minimum sampling frequency from low pass
    consideration is '+string(S_LOW)+' Hz')
15
16 //B = fH-fL = 2000 Hz
17 B = fH-fL
```

```

18
19 //k = floor(fH/B) = 2, where floor(x) gives the
    largest integer that does not exceed x
20 k = floor(fH/B)
21
22 //The required sampling frequency from band pass
    consideration(S_BAND) = 2*fH/k
23 S_BAND = 2*fH/k
24
25 disp('Minimum sampling frequency from band pass
    consideration is '+string(S_BAND)+' Hz')

```

---

#### Scilab code Exa 5.3 Calculation of Guard time

```

1  clc;
2  //page 259
3  //problem 5.3
4
5  //Given width of each pulse W = 150 us
6  W = 150 * 10^-6
7
8  //One cycle is a period ,T = 1ms
9  T = 1000 * 10^-6
10
11 //There are 5 messages multiplexed each
    utilizeallocated time pulse width = s(T_5) = T/5
12 T_5 = T/5
13
14 //Gaurd time(GT_5) = allocated time-pulse width =
    T_5-W
15 GT_5 = T_5-W
16
17 disp('Gaurd time where 5 messages multiplexed is '+
    string(GT_5)+' seconds')
18

```

```

19 //Here there are 10 messages multiplexed each
    utilizeallocated time pulse width =  $s(T_{10}) = T$ 
    /10
20 T_10 = T/10
21
22 //Gaurd time(GT_10) = allocated time-pulse width =
    T_10-norrow pulses width =  $T_{10} - 50 * 10^{-6}$ 
23 GT_10 = T_10 - 50 * 10^-6
24
25 disp('Gaurd time where 10 messages multiplexed is '+
    string(GT_10)+' seconds ')

```

---

**Scilab code Exa 5.5** Calculation of minimum number of Binary digits

```

1  clc;
2  //page 272
3  //problem 5.5
4
5  //Let Abe the maximum value of the discrete samples.
6  //Error tolerated is 0.1% i.e. 0.001A
7  //If D is step size then possible maximum error is D
    /2
8  //Thus  $D/2 = 0.001A$  or  $A/D = 500 =$  no of levels
    required(Levels)
9  Levels = 500
10
11 //minimum no of binary digits required(B) = rounded
    value to the next higher integer of log2 (Levels)
12 B = round(log2 (Levels))
13
14 disp('Minimum no of binary digits required '+string(
    B))

```

---

### Scilab code Exa 5.6 Companding

```
1  clc ;
2  //page 273
3  //problem 5.6
4
5  //The y axis is uniformly quantized with step size(
   step_size) = 1/[(2^8)/(2-1)] in both +ve & -ve
   direction between 1 & -1 when peak of input
   varies between 1 & -1.
6  //The smallest step in x direction occurs nearest to
   x=0 i.e between y1 = 0 & y2 = step_size
7  step_size = 1/[(2^8)/2-1]
8  y1 = 0
9  y2 = step_size
10
11 //Then, y1 = [ln(1+255*x1)]/[ln(1+255)]
12
13 x1 = (%e^(y1*log(256)) - 1)/255;
14
15 //y2 = [ln(1+255*x2)]/[ln(1+255)]
16 x2 = (%e^(y2*log(256)) - 1)/255;
17
18 //The smallest step size is 10*(x2-x1)
19 disp('The smallest step size is '+string(10*(x2-x1))
   +' Volts')
20
21 //The largest step size occurs when x is at its
   extreme between y1 = 1-1/127 = 126/127 & y2 = 1
22 y1 = 1-1/127
23 y2 = 1
24
25 //Then, y1 = [ln(1+255*x1)]/[ln(1+255)]
26
27 x1 = (%e^(y1*log(256)) - 1)/255;
28
29 //y2 = [ln(1+255*x2)]/[ln(1+255)]
30 x2 = (%e^(y2*log(256)) - 1)/255;
```



```

31
32 //The largest step size is 10*(x2-x1)
33 disp('The largest step size is '+string(10*(x2-x1))+
      ' Volts')

```

---

### Scilab code Exa 5.9 LMS Algorithm

```

1  clc;
2  //page 296
3  //problem 5.9
4
5  //for error calculation  $e(n) = m(n) - [\hat{h}_j(n)*m(n-1)$ 
    $+ \hat{h}_j(n)*m(n-2) + \hat{h}_j(n)*m(n-3) + \dots + \hat{h}_j(n)*m(n$ 
    $-N]$ 
6
7  //for coefficient upgradation  $\hat{h}_j(n+1) = \hat{h}_j(n) + u(m(n$ 
    $-j)e(n)$  where  $u =$  learning parameter  $= 0.1$ .
8   $u = 0.1$ 
9
10 //Assign m values taking from  $m = -3$  to 5
11 //Denoting  $m(x)$  as matrix m where each element
   represents from  $n = -3$  to 5
12  $m = [0 \ 0 \ 0 \ 1 \ 2 \ 3 \ 4 \ 5 \ 6]$ 
13
14 //taking  $e(n)$  as matrix e,  $\hat{h}_j(n)$  as matrices  $h_j$ 
15  $e = \mathbf{zeros}(1,5)$ 
16  $h_1 = \mathbf{zeros}(1,6)$ 
17  $h_2 = \mathbf{zeros}(1,6)$ 
18
19 //given  $\hat{h}_1(0) = \hat{h}_2(0) = 0$ 
20  $h_1(1) = 0$ 
21  $h_2(1) = 0$ 
22
23 for  $i = 1:5$ 
24      $e(i) = m(i+3) - h_1(i)*m(i+2) - h_2(i)*m(i+1)$ 

```

```

25     h_1(i+1) = h_1(i) + u*m(i+2)*e(i)
26     h_2(i+1) = h_2(i) + u*m(i+1)*e(i)
27 end
28
29 //here e(3) is given as 1.32 but it is displaying
    0.92
30 //here ^h2(3) is given as 0.26 but it is displaying
    0.46
31
32 for i = 1:5
33     disp('e(' + string(i-1) + ') = ' + string(e(i)))
34     disp('^h1(' + string(i) + ') = ' + string(h_1(i+1)))
35     disp('^h2(' + string(i) + ') = ' + string(h_2(i+1)))
36 end

```

---

#### Scilab code Exa 5.11 SNR of a DM system

```

1  clc;
2  //page 296
3  //problem 5.11
4
5  //case 1(a)
6  //f = 400Hertz , fs = 8000Hertz
7  f = 400
8  fs = 8000
9
10 //We know that maximum signal to noise ratio (SNR_max
    ) = 3*(fs^2)/(8*(pi^2)*(f^2))
11 SNR_max = 3*(fs^2)/(8*(%pi^2)*(f^2))
12 //SNR_max in decibels is SNR_max_db
13 SNR_max_db = 10*log10 (SNR_max)
14
15 disp('Maximum signal to noise ratio for f = 400 & fs
    = 8000 is ' + string(SNR_max) + ' = ' + string(
    SNR_max_db) + ' db')

```

```

16
17 //case 1(b)
18 //f = 400Hertz , fs = 16000Hertz
19 f = 400
20 fs = 16000
21
22 //We know that maximum signal to noise ratio(SNR_max
    ) =  $3*(fs^2)/(8*(pi^2)*(f^2))$ 
23 SNR_max =  $3*(fs^2)/(8*(pi^2)*(f^2))$ 
24
25 //SNR_max in decibels is SNR_max_db
26 SNR_max_db =  $10*\log_{10}(SNR\_max)$ 
27
28 //Given solution is 13.8385 dB obtained solution is
    17.838515 dB
29
30 disp('Maximum signal to noise ratio for f = 400 & fs
    = 16000 is '+string(SNR_max)+' = '+string(
    SNR_max_db)+' db')
31
32 //case 2(a)
33 //f = 400Hertz , fs = 8000Hertz & fc = 1000Hertz
34 f = 400
35 fs = 8000
36 fc = 1000
37
38 //If a 1kHz low pass post reconstruction filter is
    used then maximum signal to noise ratio(SNR_max)
    =  $3*(fs^3)/(8*(pi^2)*(f^2)*fc)$ 
39 SNR_max =  $3*(fs^3)/(8*(pi^2)*(f^2)*fc)$ 
40 //SNR_max in decibels is SNR_max_db
41 SNR_max_db =  $10*\log_{10}(SNR\_max)$ 
42
43 disp('If a 1kHz low pass post reconstruction filter
    is used then')
44
45 disp('Maximum signal to noise ratio for f = 400, fs
    = 8000 & fc = 1000 is '+string(SNR_max)+' = '+

```

```
    string(SNR_max_db)+' db')
46
47 //case 2(b)
48 //f = 400Hertz, fs = 16000Hertz & fc = 1000Hertz
49 f = 400
50 fs = 16000
51 fc = 1000
52
53 //If a 1kHz low pass post reconstruction filter is
    used then maximum signal to noise ratio(SNR_max)
    = 3*(fs^3)/(8*(pi^2)*(f^2)*fc)
54 SNR_max = 3*(fs^3)/(8*(%pi^2)*(f^2)*fc)
55 //SNR_max in decibels is SNR_max_db
56 SNR_max_db = 10*log10 (SNR_max)
57
58 disp('Maximum signal to noise ratio for f = 400, fs
    = 16000 & fc = 1000 is '+string(SNR_max)+' = '+
    string(SNR_max_db)+' db')
```

---

# Chapter 6

## Digital Modulation and Transmission

Scilab code Exa 6.1 Carrier phase variation

```
1  clc ;
2  //page 341
3  //problem 6.1
4
5  //Given messages signal m = [1,0,1,1,0,1]
6  m = [1,0,1,1,0,1];
7
8  //Logical 0 corresponds to pi i.e %pi and Logical 1
   corresponds to 0
9
10 //
    //////////////////////////////////////
11
12 //For BPSK, from the above deduction let the carrier
   phase be Carrier_Phase_BPSK
13 for i = 1:5
14     if m(i)==1 then
15         Carrier_Phase_BPSK(i) = 0;
```

```

16     else
17         Carrier_Phase_BPSK(i) = %pi;
18     end
19 end
20
21 disp(Carrier_Phase_BPSK, 'The Phase of the carrier
    signal for BPSK varies as ');
22
23 //
    //////////////////////////////////////
24
25 //For DPSK
26 //Let b represent the input to balance modulator
27
28 //If the initial value of b be 0
29 b = 0;
30
31 for i = 2:5
32     b(i) = bitxor(m(i),b(i-1))
33 end
34
35 //Now the carrier phase, Carrier_Phase_DPSK
36 for i = 1:5
37     if b(i)==1 then
38         Carrier_Phase_DPSK(i) = 0;
39     else
40         Carrier_Phase_DPSK(i) = %pi;
41     end
42 end
43
44 //Now the carrier amplitude, Carrier_Amplitude_DPSK
45 for i = 1:5
46     Carrier_Amplitude_DPSK(i) = cos(
        Carrier_Phase_DPSK(i));
47 end
48
49 disp(Carrier_Phase_DPSK, 'The Phase of the carrier

```

```

        signal for DPSK varies as follows, '+'when the
        initial value of b is 1');
50 disp(Carrier_Amplitude_DPSK, 'The Amplitude of the
        carrier signal for DPSK varies as follows, '+'
        when the initial value of b is 1');
51
52 //If the initial value of b be 1
53 b = 1;
54
55 for i = 2:5
56     b(i) = bitxor(m(i),b(i-1))
57 end
58
59 //Now the carrier phase, Carrier_Phase_DPSK
60 for i = 1:5
61     if b(i)==1 then
62         Carrier_Phase_DPSK(i) = 0;
63     else
64         Carrier_Phase_DPSK(i) = %pi;
65     end
66 end
67
68 //Now the carrier amplitude, Carrier_Amplitude_DPSK
69 for i = 1:5
70     Carrier_Amplitude_DPSK(i) = cos(
        Carrier_Phase_DPSK(i));
71 end
72
73 disp(Carrier_Phase_DPSK, 'The Phase of the carrier
        signal for DPSK varies as follows, '+'when the
        initial value of b is 0');
74 disp(Carrier_Amplitude_DPSK, 'The Amplitude of the
        carrier signal for DPSK varies as follows, '+'
        when the initial value of b is 0');
75
76 //
        //////////////////////////////////////

```

```

77
78 //For DEPSK
79 //The DEPSK transmitter output is same as that of
    DPSK
80
81 //If the initial value of b be 0
82 b = 0;
83
84 for i = 2:5
85     b(i) = bitxor(m(i),b(i-1))
86 end
87
88 //Now the carrier phase, Carrier_Phase_DPSK
89 for i = 1:5
90     if b(i)==1 then
91         Carrier_Phase_DEPSK(i) = 0;
92     else
93         Carrier_Phase_DEPSK(i) = %pi;
94     end
95 end
96
97 disp(Carrier_Phase_DEPSK,'The Phase of the carrier
    signal for DEPSK varies as follows, '+'when the
    initial value of b is 1');
98
99 //If the initial value of b be 1
100 b = 1;
101
102 for i = 2:5
103     b(i) = bitxor(m(i),b(i-1))
104 end
105
106 //Now the carrier phase, Carrier_Phase_DPSK
107 for i = 1:5
108     if b(i)==1 then
109         Carrier_Phase_DEPSK(i) = 0;
110     else
111         Carrier_Phase_DEPSK(i) = %pi;

```



```

112     end
113 end
114
115 disp(Carrier_Phase_DEPSK, 'The Phase of the carrier
    signal for DEPSK varies as, '+' when the initial
    value of b is 0');

```

---

### Scilab code Exa 6.2 DPSK and DEPSK

```

1  clc;
2  //page 341
3  //problem 6.2
4
5  //From Ex6_1 the obtained carrier amplitude is c
6
7  //
   ///////////////////////////////////////////////////////////////////
8
9  //For DPSK
10 //Considering the initial value of the storage
    element to be 0 in polar and -1 in bipolar
11 c = [1,1,-1,1,1];
12 y = -1;
13 //Let the output be y
14 for i = 2:5
15     y(i) = c(i)*c(i-1);
16 end
17
18 //Converting back to binary data
19 for i = 1:5
20     if y(i)== -1 then
21         output_binary(i) = 0;
22     else
23         output_binary(i) = 1;

```



```

5 //Given energy per bit Eb = 0.01
6 Eb = 0.01;
7
8 //Given fundamental frequency is fb = 8 KHz
9 fb = 8*10^3;
10
11 //No of symbols M = 16
12 M = 16;
13
14 N = log2(M);
15
16 BW_BPSK = 2*fb;
17 disp('Bandwidth for BPSK is '+string(BW_BPSK)+'Hz');
18
19 BW_QPSK = fb;
20 disp('Bandwidth for QPSK is '+string(BW_QPSK)+'Hz');
21
22 BW_16MPSK = fb/2;
23 disp('Bandwidth for 16 MPSK is '+string(BW_16MPSK)+'
    Hz');
24
25 BW_BFSK = 4*fb;
26 disp('Bandwidth for BFSK is '+string(BW_BFSK)+'Hz');
27
28 BW_MSK = 1.5*fb;
29 disp('Bandwidth for MSK is '+string(BW_MSK)+'Hz');
30
31 BW_16MFSK = 2*M*fb;
32 disp('Bandwidth for 16 MFSK is '+string(BW_16MFSK)+'
    Hz');
33
34
35 Min_dist_BPSK = 2*(Eb)^0.5;
36 disp('Minimum distance in signal space in BPSK is '+
    string(Min_dist_BPSK));
37
38 Min_dist_QPSK = 2*(Eb)^0.5;
39 disp('Minimum distance in signal space in QPSK is '+

```

```

    string(Min_dist_QPSK));
40
41 //The given answer in the textbook is 0.0152, which
    appears to be wrong. The correct answer is 0.078
42 Min_dist_16MPSK = (4*N*Eb*(sin(%pi/16))^2)^0.5;
43 disp('Minimum distance in signal space in 16 MPSK is
    '+string(Min_dist_16MPSK));
44
45 Min_dist_BFSK = (2*Eb)^0.5;
46 disp('Minimum distance in signal space in ortho BFSK
    is '+string(Min_dist_BFSK));
47
48 Min_dist_MSX = 2*(Eb)^0.5;
49 disp('Minimum distance in signal space in MSK is '+
    string(Min_dist_MSX));
50
51 Min_dist_16MFSK = (2*N*Eb)^0.5;
52 disp('Minimum distance in signal space in ortho 16
    MFSK is '+string(Min_dist_16MFSK));
53
54 disp('The best method that provides least noise
    susceptibility is 16 MFSK, then BPSK, then QPSK,
    then comes MSK, then orthogonal BFSK and finally
    16 MPSK')

```

---

### Scilab code Exa 6.5 Duobinary decoding

```

1 clc;
2 //page 381
3 //problem 6.5
4
5 //Given input signal is d
6 d = [0,1,1,1,0,1,0,1,1];
7
8 //

```

```

////////////////////////////////////
9 //The answers obtained here are different from the
   ones mentioned in the textbook.
10 //The given answers have been checked rigorously and
   have been found out to be true.
11
12 //When precoded
13
14 //Signal b is initially assumed to be 0
15 b = 0;
16
17 for i = 2:9
18     b(i) = bitxor(b(i-1),d(i));
19 end
20
21 //Changing bit code to polar signal we get, 0 -->
   -1, 1 --> +1
22 for i = 1:9
23     if b(i)==1 then
24         bp(i) = 1;
25     else
26         bp(i) = -1;
27     end
28 end
29
30 //Let initial value of Vd be 0
31 //Vd = 0;
32
33 for i = 2:9
34     Vd(i) = bp(i) + bp(i-1);
35 end
36
37 //Converting polar signal to bit code we get, -2 -->
   0, 0 --> 1, 2 --> 0
38 for i = 1:9
39     if Vd(i)== -2 then
40         da(i) = 0;

```

```

41     elseif Vd(i)== 2 then
42         da(i) = 0;
43     else
44         da(i) = 1;
45     end
46 end
47
48 disp(da, 'Decoded output when precoded is ')
49
50 //
    //////////////////////////////////////
51
52 //When not precoded exor gate is not there
53
54 //Changing bit code to polar signal we get , 0 -->
    -1, 1 --> +1
55 for i = 1:9
56     if d(i)==1 then
57         dp(i) = 1;
58     else
59         dp(i) = -1;
60     end
61 end
62
63 for i = 2:9
64     Vd(i) = dp(i) + dp(i-1);
65 end
66
67 //Converting polar signal to bit code we get , -2 -->
    0, 0 --> 1, 2 --> 1
68 for i = 2:9
69     if Vd(i)== -2 then
70         da(i) = 0;
71     elseif Vd(i)== 2 then
72         da(i) = 0;
73     else
74         da(i)= ~da(i-1);

```

```
75     end
76 end
77
78 disp(da, 'Decoded output when not precoded is ')
```

---

#### Scilab code Exa 6.6 Roll off factor

```
1  clc;
2  //page 381
3  //problem 6.6
4
5  //Given Bandwidth BW = 4 kHz
6  BW = 4*10^3;
7
8  //Given data rate is fb = 6 kbps
9  fb = 6*10^3;
10
11 //The roll off factor alpha is
12 alpha = ((2*BW)/fb) - 1;
13
14 disp('The roll off factor is '+string(alpha));
15
16 //
17  ///////////////////////////////////////////////////
18 //The required data rate supported at alpha = 0.25
19   is D
19 alpha = 0.25
20
21 //The corresponding expression for D is
22 D = (2*BW)/(1+alpha);
23
24 disp('The supported data rate is '+string(D)+' kbps'
25      ');
```

```
25
26 //For full roll-off alpha = 1.0,
27 alpha = 1;
28
29 fb = 2*BW/(1+alpha);
30
31 disp('The data rate is '+string(fb)+' kbps');
```

---



# Chapter 7

## Mathematical Representation of Noise

Scilab code Exa 7.3 Noise Power

```
1  clc;
2  //page 413
3  //problem 7.3
4
5  //The resistance R = 1000 Ohm
6  R = 10^3;
7
8  //The capacitance C = 0.5*10^-6 F
9  C = 0.1*10^-6;
10
11 //Cutoff frequency for RC filter is f
12 f = 1/(2*pi*R*C)
13
14 //White noise power spectral density n
15 n = 10^(-9);
16
17 //Noise power at filter output P
18 P = (pi/2)*n*f;
19
```

```

20 disp('Noise power at output filter is '+string(P)+'
      Watt');
21
22 //Noise power at filter output P_new when cutoff
      frequency is doubled
23 P_new = (%pi/2)*n*2*f;
24
25 disp('Noise power at output filter when cutoff
      frequency is doubled is '+string(P_new)+' Watt');
26
27 //Ideal Low Pass filter Bandwidth B = 1000 Hz
28 B = 1000;
29
30 disp('Output Noise Power is '+string(n*B)+' Watt');
31
32 disp('Output Noise Power when cut-off frequency is
      doubled is '+string(2*n*B)+' Watt');
33
34 //Proportionality constant T = 0.01
35 T = 0.01;
36
37 //Output noise power O
38 O = n*(B^3)*(T^2)*(4/3)*(%pi)^2;
39
40 disp('Output Noise Power when signal is passed
      through a differentiator passed through ideal low
      pass filter '+string(O)+' Watt');
41
42 O_new = 8*n*(B^3)*(T^2)*(4/3)*(%pi)^2;
43
44 disp('Output Noise Power when signal is passed
      through a differentiator passed through ideal low
      pass filter and when cut-off frequency is
      doubled is '+string(O_new)+' Watt');

```

---

Scilab code Exa 7.4 SNR at output of equalizer

```
1  clc;
2  //page 413
3  //problem 7.4
4
5  //Given signal strength S = 0.001 W
6  S = 0.001;
7
8  //Gaussian Noise Magnitude n
9  n = 10(-8);
10
11 //Frequency of signal f = 4000 Hz
12 F = 4000;
13
14 //Noise at equalizer output N
15 N = integrate('n*(1+(f^2)/F^2)', 'f', -F, F);
16
17 //Signal to Noise Ratio value is SNR
18 SNR = S/N;
19
20 disp('SNR value is '+string(10*log10(SNR))+ ' dB');
```

---

# Chapter 8

## Noise in Amplitude Modulation System

Scilab code Exa 8.1 SNR of SSB signal

```
1  clc;
2  //page 436
3  //problem 8.1
4
5  //Given frequency range fc= 1MHz to fc = 1.0005Mhz
6  //Single side message bandwidth is fM
7  fM= (1.0005 - 1)*10^6;
8  disp('Message bandwidth is '+string(fM)+' Hz');
9  //The textbook contains a calculation error here.
   The calculated fM in the textbook is 500kHz
   instead of 5kHz, following which all the
   solutions are erroneous
10
11 //Given input signal strength Si= 1mW
12 //Let output signal strength be So
13 //So=Si/4
14 Si= 10^(-3);
15 So= Si/4;
16 disp('Signal output strength is '+string(So)+' dB');
```

```

17
18 //Given Power Spectral Density n = 10-9 W/Hz
19 //Let output noise strength be No
20 n= 10-9;
21 No= (n*fM)/4;
22 disp('Output Noise Strength is '+string(No)+' dB');
23
24 //Let SNR at filter output be SNR
25 SNR= So / No;
26 disp('Output SNR is '+string(SNR)+' dB');
27
28 //By reduction of message signal Bandwidth the
    Output Noise strength changes
29 //Let the new output noise strength, bandwidth and
    SNR be be No_new, fM_new and SNR_new respectively
30 fM_new = 75/100*fM;
31 No_new = n*fM_new/4;
32 SNR_new = So / No_new;
33
34 disp('Changed SNR is '+string(SNR_new)+' dB');

```

---

### Scilab code Exa 8.2 Signal strength and noise power density

```

1  clc;
2  //page 436
3  //problem 8.2
4
5  //Given frequency range fc - fm = 0.995MHz to fc +
    fm = 1.005Mhz
6  //Double side message bandwidth is fM
7  fM= (1.005 - 0.995)*106 / 2;
8  disp('Message bandwidth is '+string(fM)+' Hz');
9  //The textbook contains a calculation error here.
10 //The calculated fM in the textbook is 500kHz
    instead of 5kHz,

```

```

11 //Following which all the solutions obtained here
    are erroneous.
12
13 //Given input signal strength Si= 1mW
14 //Let output signal strength be So
15 //So=Si/2
16 Si= 10(-3);
17 So= Si/2;
18 disp('Signal output strength is '+string(So)+' dB');
19
20 //Given Power Spectral Density n = 10-9 W/Hz
21 //Let output noise strength be No
22 n= 10-9;
23 No= (n*fM)/2;
24 disp('Output Noise Strength is '+string(No)+' dB');
25
26 //Let SNR at filter output be SNR
27 SNR= So / No;
28 disp('Output SNR of the DSB-SC wave is '+string(SNR)
    +' dB');
29
30 //By reduction of message signal Bandwidth the
    Output Noise strength changes
31 //Let the new output noise strength, bandwidth and
    SNR be be No_new, fM_new and SNR_new respectively
32 fM_new = 75/100*fM;
33 No_new = n*fM_new/4;
34 SNR_new = So / No_new;
35 disp('Changed SNR is '+string(SNR_new)+' dB');

```

---

### Scilab code Exa 8.3 Minimum transmitter power

```

1 clc;
2 //page 446
3 //problem 8.3

```

```

4
5 //Given bandwidth of signal is fM = 4kHz
6 fM = 4*10^3;
7 //Given power spectral density of white noise n =
      2*10^-9 W/Hz
8 n = 2*10^-9;
9 //Also given that minimum output SNR is 40dB
10 //Signal undergoes a loss of 30dB
11
12 //For SSB:
13 // Required minimum output SNR = Si_min_SSB / (n*fM)
      = 40 dB = 10^4
14 Si_min_SSB = (10^4)*n*fM;
15 // Required minimum signal strength at transmitter
      output Si_tran = Si_min * 30 dB
16 Si_tran_SSB = Si_min_SSB * 10^3;
17 disp('Required minimum SSB signal strength at
      transmitter output is '+string(Si_tran_SSB)+' W');
18
19 //For DSB-SC:
20 // Required minimum output SNR = (Si_min_DSB/3) / (n
      *fM) = 40 dB = 10^4
21 Si_min_DSB = 3*(10^4)*n*fM;
22 // Required minimum signal strength at transmitter
      output Si_tran = Si_min * 30 dB
23 Si_tran_DSB = Si_min_DSB * 10^3;
24 disp('Required minimum DSB signal strength at
      transmitter output is '+string(Si_tran_DSB)+' W');

```

---

#### Scilab code Exa 8.4 SNR of a Square Demodulator

```

1 clc;
2 //page 447
3 //problem 8.4
4

```

```

5 //Given bandwidth of signal is fM = 60 kHz
6 fM = 60*10^3;
7
8 //Given power spectral density of white noise n =
   2*10^-6 W/Hz
9 n = 2*10^-6;
10
11 //Given time average of square of mssg signal P =
   0.1W
12 P = 0.1;
13
14 //Noise power at input baseband range NM
15 NM = n * fM;
16
17 //Threshold occurs at carrier power Pc = 2.9 * NM
18 Pc_Threshold = 2.9 * NM;
19
20 //For carrier power Pc = 10W, output SNR
21 Pc = 10;
22 SNRo = Pc * P / NM ;
23 disp('Output SNR is '+string(SNRo)+' dB');
24
25 //Carrier power is reduced by 100 times making the
   new power Pc_new
26 Pc_new = Pc / 100;
27
28 //In the given solutions the NM value is 1.2W
   instead of 0.12W
29 //The corect answer is 0.0925926 instead of 0.000926
30 SNR_new = (4/3) * P * (Pc_new/NM)^2;
31 disp('Output SNR when carrier power is reduced is '+
   string(SNR_new)+' dB');

```

---



# Chapter 9

## Noise in Frequency Modulation Systems

Scilab code Exa 9.1 SNR of an FM Limiter Discriminator Demodulator

```
1  clc;  
2  //page 463  
3  //problem 9.1  
4  
5  //Input signal strength Si = 0.5 W  
6  Si = 0.5;  
7  
8  //Gaussian Power Spectral Density n = 10(-10) W/Hz  
9  n = 10(-10);  
10  
11 //Baseband cutoff signal fM = 15 kHz  
12 fM = 15 * 103;  
13  
14 //Maximum frequency deviation Df = 60 kHz  
15 Df = 60 * 103;  
16  
17 //Average power of the modulating signal mt = 0.1 W  
18 mt = 0.1;  
19
```

```

20 SNR = (3/(4*%pi^2))*((Df/fM)^2)*mt^2*(Si/(n*fM));
21
22 disp('SNR is '+string(10*log10(SNR))+ ' dB');
23
24 //Part b
25
26 //Required SNR at output >40 dB = 10000
27
28 //From (a), required Si/0.5 > 10000/4052.8
29 //Or, required Si > 1.2337 W
30 //Since, channel loss is 20 dB (=100),
31 //Required transmitter power > 1.2337*100 = 123.37
32
33 disp('Required transmitter power > 1.2337 x 100 =
    123.37 ');

```

---

#### Scilab code Exa 9.2 FM Limiter Discriminator Demodulator

```

1  clc;
2  //page 464
3  //problem 9.2
4
5  //Baseband cutoff signal fM = 15 kHz
6  fM = 15 * 10^3;
7
8  //Maximum frequency deviation Df = 60 kHz
9  Df = 60 * 10^3;
10
11 //Figure of Merit for FM is G_FM
12 G_FM = (3/2)*(Df/fM)^2;
13
14 disp('Figure of Merit for FM system is '+string(G_FM
    ));
15
16 //Ratio of Figure of Merits of FM and AM systems is

```

```

R
17 R = G_FM/(1/3);
18
19 disp('Ratio of Figure of Merits for FM and AM
      systems is '+string(R));
20
21 Df_new = 2*Df;
22
23 //Figure of Merit for FM when bandwidth is doubled
      is G_FM_new
24 G_FM_new = (3/2)*(Df_new/fM)^2;
25
26 //Ratio of Figure of Merits of FM and AM systems
      when bandwidth is doubled is R_new
27 R_new = G_FM_new/(1/3);
28
29 disp('Ratio of Figure of Merits for FM and AM
      systems when bandwidth is doubled is '+string(
      R_new));

```

---

### Scilab code Exa 9.3 RC Filter Preemphasis Deemphasis

```

1  clc;
2  //page 475
3  //problem 9.3
4
5  //Resistance R = 1000 Ohm
6  R = 10^3;
7
8  //Capacitance C = 0.1 * 10^-6 F
9  C = 0.1*10^-6;
10
11 //Break point for RC filter is f1
12 f1 = 1/(2*%pi*R*C)
13

```

```

14 //Baseband bandwidth of signal fM = 15 kHz
15 fM = 15 * 10^3;
16
17 Gain = atan(fM/f1)/(3*(f1/fM)*[1 - (f1/fM)*atan(fM/
    f1)]);
18
19 disp('Initial Gain is '+string(10*log10(Gain))+ ' dB'
    );
20
21 //New Baseband bandwidth of signal fM_new = 15 kHz
22 fM_new = 2*15 * 10^3;
23
24 Gain_new = atan(fM_new/f1)/(3*(f1/fM_new)*[1 - (f1/
    fM_new)*atan(fM_new/f1)]);
25
26 disp('Final Gain is '+string(10*log10(Gain_new))+ '
    dB');

```

---

#### Scilab code Exa 9.6 SNR at Input

```

1  clc;
2  //page 495
3  //problem 9.6
4
5  //Baseband cutoff signal fM = 15 kHz
6  fM = 15 * 10^3;
7
8  //Carrier filter bandwidth is B = 60 kHz
9  B = 60 * 10^3;
10
11 //RMS frequency division Df_RMS = 30 kHz
12 Df_RMS = 30 * 10^3;
13
14 //Let a = Df_RMS/fM for substitution
15 a = Df_RMS/fM;

```

```

16
17 //Let b = fM/B for substitution
18 b = fM/B;
19
20 //Let input SNR 1 be I_SNR1 = 10 dB = 10
21 I_SNR1 = 10;
22
23 //Output SNR is O_SNR1
24 O_SNR1 = (3*(a^2)*I_SNR1)/(1+6*((2/%pi)^0.5)*I_SNR1*
    exp(-(b)*I_SNR1));
25
26 disp('Output SNR is '+string(10*log10(O_SNR1))+ ' dB'
    );
27
28 //Let input SNR 2 be I_SNR2 = 20 dB = 100
29 I_SNR2 = 100;
30
31 //Output SNR is O_SNR2
32 O_SNR2 = (3*(a^2)*I_SNR2)/(1+6*((2/%pi)^0.5)*I_SNR2*
    exp(-(b)*I_SNR2));
33
34 //Solution given in the book is 13.5431 which is
    fallacious , the correct answer is 24.32444
35 disp('Output SNR is '+string(10*log10(O_SNR2))+ ' dB'
    );
36
37 //Let input SNR 3 be I_SNR3 = 30 dB = 1000
38 I_SNR3 = 1000;
39
40 //Output SNR is O_SNR3
41 O_SNR3 = (3*(a^2)*I_SNR3)/(1+6*((2/%pi)^0.5)*I_SNR3*
    exp(-(b)*I_SNR3));
42
43 disp('Output SNR is '+string(10*log10(O_SNR3))+ ' dB'
    );

```

---

# Chapter 10

## Phase Locked Loops

Scilab code Exa 10.3 SNR of Phase Discriminator

```
1  clc;
2  //page 520
3  //problem 10.3
4
5  // Part (a)
6
7  //Input SNR SNR_ip
8  SNR_ip = 1000;
9
10 //Beta B
11 B = 10;
12
13 //Output SNR SNR_op
14 SNR_op = (1.5*(B^2)*SNR_ip)/(1 + (12*B/%pi)*(SNR_ip)
15         *exp(-0.5*(1/(B+1))*(SNR_ip)));
16 disp('Output SNR is '+string(10*log10(SNR_op))+ ' dB
17     ');
18 // Part (b)
19
```

```

20 //Input SNR SNR_ip
21 SNR_ip = 10;
22
23 //Output SNR SNR_op
24 SNR_op = (1.5*(B^2)*SNR_ip)/(1 + (12*B/%pi)*(SNR_ip)
      *exp(-0.5*(1/(B+1))*(SNR_ip)));
25
26 disp('Output SNR is '+string(10*log10(SNR_op))+ ' dB
      ');

```

---

#### Scilab code Exa 10.5 Channel Spacing

```

1  clc;
2  //page 533
3  //problem 10.5
4
5  //Given reference frequency is fref = 10 MHz
6  fref = 10 * 10^6;
7
8  //Given step frequency is fstep = 100 KHz
9  fstep = 100 * 10^3;
10
11 //Division ratio M
12 M = fref/fstep;
13
14 //Required output frequency F = 100.6 MHz
15 F = 100.6 * 10^6;
16
17 N = F/fstep;
18
19 //Given P = 64
20 P = 64;
21
22 //Truncating value B = 15
23 B = 15;

```

```

24
25 A = N - P*B;
26
27 disp('The value of A is '+string(A));
28 disp('The value of B is '+string(B));
29 disp('The value of M is '+string(M));

```

---

### Scilab code Exa 10.7 Phase Locked Loops

```

1  clc;
2  //page 534
3  //problem 10.7
4
5  //Given reference frequency for PLL is fref = 0.48
   MHz
6  fref = 0.48 * 10^6;
7
8  //Frequency divider N = 2000
9  N = 2000;
10
11 //Output Frequency fout
12 fout = fref*N;
13
14 //Output Frequency favg
15 favg = (2000*15 + 2001*1)*(0.48/16) * 10^6;
16
17 disp('Output frequency is '+string(fout)+' Hz');
18
19 //Reference frequency is not subdivided before going
   to comparator and it is an integer divider in
   the feedback path the frequency resolution fres =
   0.48 * 10^6;
20 fres = 0.48 * 10^6;
21
22 disp('Frequency resolution is '+string(fres)+' Hz');

```



23

```
24 disp('Output frequency resolution is '+string(favg -  
      fout)+' Hz');
```

---

# Chapter 11

## Optimal Reception of Digital Signal

Scilab code Exa 11.3 Optimal threshold and probability of error

```
1  clc;  
2  //page 558  
3  //problem 11.3  
4  
5  //Prior probability of s1 P_s1 = 0.4  
6  P_s1 = 0.4;  
7  
8  //Prior probability of s2 P_s2 = 1 - P_s1  
9  P_s2 = 1 - P_s1;  
10  
11 //Voltage level V1 = 1  
12 V1 = 1;  
13  
14 //Voltage level V2 = -1  
15 V2 = -1;  
16  
17 //Part a  
18  
19 //Noise Variance sigma1 = 10^-3
```

```

20 sigma1 = 10^-3;
21
22 //Decision Threshold lambda1
23 lambda1 = (V1+V2)/2 + (sigma1)*log(P_s2/P_s1)/(V1-V2
    );
24
25 //Probability of error Pe
26 Pe = 0.5*(2*P_s1 - P_s1*erfc(((V2-V1)/(2*sigma1
    *2^0.5)) + (sigma1)*log(P_s2/P_s1)/((V1-V2)
    *2^0.5)));
27
28 disp('The decision threshold is '+string(lambda1)+'
    V');
29 disp('The probability of error is approximately '+
    string(Pe));
30
31 //Part b
32
33 //Noise Variance sigma2 = 10^-1
34 sigma2 = 10^-1;
35
36 //Decision Threshold lambda2
37 lambda2 = (V1+V2)/2 + (sigma2)*log(P_s2/P_s1)/(V1-V2
    );
38
39 //Probability of error Pe
40 Pe1 = 0.5*(2*P_s1 - P_s1*erfc(((V2-V1)/(2*sigma2
    *2^0.5)) + (sigma2)*log(P_s2/P_s1)/((V1-V2)
    *2^0.5)));
41
42 //In the textbook Pe has been calculated to be 0.0021
    because of the use of a very high precision
    calculator, unfortunately in scilab the function
    erfc approximates the output value to a larger
    extent due to which an exact value cannot be
    obtained.
43
44 disp('The decision threshold is '+string(lambda2)+'

```

```
    V');  
45 disp('The probability of error is approximately '+  
    string(Pe1));
```

---

#### Scilab code Exa 11.4 Decision threshold

```
1  clc;  
2  //page 559  
3  //problem 11.4  
4  
5  //Part b  
6  
7  //Voltage level V1 = 1  
8  V1 = 1;  
9  
10 //Voltage level V2 = -1  
11 V2 = -1;  
12  
13 //Prior probability of s1 P_s1 = 0.4  
14 P_s1 = 0.4;  
15  
16 //Prior probability of s2 P_s2 = 1 - P_s1  
17 P_s2 = 1 - P_s1;  
18  
19 //Cost of selecting s1 when s2 is transmitted C12 =  
    0.7  
20 C12 = 0.7;  
21  
22 //Cost of selecting s2 when s1 is transmitted C21 =  
    1 - C12  
23 C21 = 1 - C12;  
24  
25 //Noise Variance sigma = 10^-3  
26 sigma = 10^-3;  
27
```

```

28 //Descision Threshold lambda
29 lambda = (V1+V2)/2 + (sigma)*log((C12*P_s2)/(C21*
      P_s1))/(V1-V2);
30
31 disp('The decision threshold is '+string(lambda)+' V
      ');

```

---

### Scilab code Exa 11.5 Probability of error of optimum filter

```

1  clc;
2  //page 567
3  //problem 11.5
4
5  //The voltage level of reciever is V = 5 mV
6  V = 5*10^-3;
7
8  //The time required to transfer one bit is T =
      1/9600 sec
9  T = 9600^-1;
10
11 //the signal energy of a bit be Es
12 Es = (V^2)*T;
13
14 //The power spectral density is n/2 = 10^-9 Watt/
      hertz
15 n = 2*10^-9;
16
17 //Probability error for optimal reciever is Pe
18 Pe = 0.5*erfc((Es/n)^0.5);
19
20 disp('The probability of error is '+string(Pe));
21
22 //When the data rate is doubled, the new effective
      energy per bit is Es_new
23 Es_new = (V^2)*(T/2);

```

```

24
25 //The new probability of error is Pe_new
26 Pe_new = 0.5*erfc((Es_new/n)^0.5);
27
28 //Percentage increase in error rate is P
29 P = 100*(Pe_new - Pe)/Pe;
30
31 disp('Percentage increase in error rate is '+string(
    P));
32
33 //Voltage required to restore probability levels ,
    V_new
34 V_new = V*2^0.5;
35
36 disp('Voltage required to restore the probability
    levels is '+string(V_new)+' Volts');

```

---

#### Scilab code Exa 11.6 Error probability of BPSK signal

```

1  clc;
2  //page 575
3  //problem 11.6
4
5  //Amplitude of signal is A = 10 mV
6  A = 10*10^-3;
7
8  //Power Spectral Density n = 2 * 10^(-9) W/Hz
9  n = 2 * 10^(-9);
10
11 //Frequency is f = 1 MHz
12 f = 1*10^6;
13
14 //Data rate is D = 10^4 bps;
15 D = 10^4;
16

```

```

17 //Time taken for a bit to traverse
18 T = 1/D;
19
20 //Energy per signal element is Es
21 Es = A^2/(2*D);
22
23 //Probability of error Pe
24 Pe = 0.5*erfc((Es/n)^0.5);
25
26 disp('Probability of error is '+string(Pe));
27
28 //Phase shift phi = %pi/6
29 phi = %pi/6;
30
31 //Probability of error Pe_local_oscillator
32 Pe_local_oscillator = 0.5*erfc(((Es/n)^0.5)*cos(phi)
    );
33
34 disp('Probability of error of local oscillator with
    phase shift is '+string(Pe_local_oscillator));
35
36 //Timing error t
37 t = 0.1*T;
38
39 //Probability of error when there is a
    synchronization fault Pe_timing_error
40 Pe_timing_error = 0.5*erfc(((Es/n)*(1 - 2*(t/T))^2)
    ^0.5);
41
42 disp('Probability of error with synchronization
    fault is '+string(Pe_timing_error));
43
44 //Probability of error when both faults occur
    Pe_both
45 Pe_both = 0.5*erfc(((Es/n)*(cos(phi)^2)*(1 - 2*(t/T)
    )^2)^0.5);
46
47 disp('Probability of error when both faults occur '+

```

```
string(Pe_both));
```

---

### Scilab code Exa 11.7 Error probability of coherent FSK signal

```
1  clc;
2  //page 575
3  //problem 11.7
4
5  //Amplitude of signal is A = 10 mV
6  A = 10*10^-3;
7
8  //Power Spectral Density n = 2 * 10^(-9) W/Hz
9  n = 2 * 10^(-9);
10
11 //Data rate is D = 10^4 bps;
12 D = 10^4;
13
14 //Time taken for a bit to traverse
15 T = 1/D;
16
17 //Energy per signal element is Es
18 Es = A^2/(2*D);
19
20 //Probability of error Pe_a
21 Pe_a = 0.5*erfc((0.6*Es/n)^0.5);
22 disp('Probability of error when offset is small is '
      +string(Pe_a));
23
24 //Probability of error Pe_b
25 Pe_b = 0.5*erfc((Es/(2*n))^0.5);
26 disp('Probability of error when frequencies used are
      orthogonal is '+string(Pe_b));
27
28 //Probability of error Pe_c
29 Pe_c = 0.5*exp(-(Es/(2*n)));
```



```
30 disp('Probability of error for non coherent
      detection is '+string(Pe_c));
```

---

### Scilab code Exa 11.8 Error probability in Optimal Reception

```
1  clc;
2  //page 588
3  //problem 11.8
4
5  //Energy associated with each bit  $E_b = 5 * 10^{(-8)}$ 
   J
6  Eb = 5 * 10(-8);
7
8  //Power Spectral Density  $n = 2 * 10^{(-9)}$  W/Hz
9  n = 2 * 10(-9);
10
11 //No of symbols M
12 M = 16
13
14 //No of bits N
15 N = log2(M);
16
17 //Error limit for 16-PSK is P_16_PSK
18 P_16_PSK = erfc(((N*Eb*(%pi)^2)/(((M)^2)*n))^0.5);
19
20 disp('Upper limit of error probability of 16 PSK
      system is '+string(P_16_PSK));
21
22 //Error limit for 16-QASK is P_16_QASK
23 P_16_QASK = 2*erfc(((0.4*Eb)/(n))^0.5);
24
25 disp('Upper limit of error probability of 16 QASK
      system is '+string(P_16_QASK));
26
27 //Error limit for 16-FSK is P_16_FSK
```

```

28 P_16_FSK = ((2^4 - 1)/2)*erfc(((N*Eb)/(2*n))^0.5);
29
30 disp('Upper limit of error probability of 16 FSK
      system is '+string(P_16_FSK)+' , negligibly small'
      );

```

---

### Scilab code Exa 11.9 Probability of error of QPR System

```

1  clc;
2  //page 595
3  //problem 11.9
4
5  //Energy associated with each bit Eb = 5 * 10^(-8)
   J
6  Eb = 5 * 10^(-8);
7
8  //Power Spectral Density n = 2 * 10^(-9) W/Hz
9  n = 2 * 10^(-9);
10
11 //Probability of error Pe
12 Pe = 0.5*erfc(((Eb*(%pi)^2)/(16*n))^0.5);
13
14 disp('Probability of error of QPR system is '+string
      (Pe));
15
16 //Given Bandwidth of channel is BW
17 BW = 10*10^3;
18
19 D = 2*BW;
20
21 disp('Data rate is '+string(D)+' bps');

```

---

# Chapter 12

## Noise in Pulse Code Modulation and Delta Modulation Systems

Scilab code Exa 12.2 SNR Optimal receiver

```
1 clc;  
2 //page 608  
3 //problem 12.2  
4  
5 //Baseband cutoff signal fM = 4 kHz  
6 fM = 4 * 103;  
7  
8 //White noise power spectral density n  
9 n = 2*10(-9);  
10  
11 // Part (a)  
12  
13 //Input Signal energy Si = 0.001  
14 Si_a = 0.001;  
15  
16 //No of levels used for PCM Coding M = 8  
17 M_a = 8;
```

```

18
19 N_a = log2(M_a);
20
21 //Input SNR is SNR_ip
22 SNR_ip = Si_a/(n*fM);
23
24 //Output SNR is SNR_op
25 SNR_op = (2^(2*N_a))/(1 + (2^(2*N_a + 1))*erfc((
    SNR_ip*(1/(2*N_a))))^0.5);
26
27 disp('Input SNR for (a) is '+string(10*log10(SNR_ip)
    )+' dB');
28 disp('Output SNR (a) is '+string(10*log10(SNR_op))+
    ' dB');
29
30 // Part (b)
31
32 //Input Signal energy Si = 0.001
33 Si_b = 0.001;
34
35 //No of levels used for PCM Coding M = 256
36 M_b = 256;
37
38 N_b = log2(M_b);
39
40 //Input SNR is SNR_ip_b
41 SNR_ip_b = Si_b/(n*fM);
42
43 //Output SNR is SNR_op_b
44 SNR_op_b = (2^(2*N_b))/(1 + (2^(2*N_b + 1))*erfc((
    SNR_ip_b*(1/(2*N_b))))^0.5);
45
46 //Unfortunately in scilab the function erfc
    approximates the output value to a larger extent
    due to which an exact value cannot be obtained.
47 //The difference in the textbook answer and obtained
    answer is significant because of converting the
    answer into dB.

```

```

48
49 disp('Input SNR for (b) is '+string(10*log10(
    SNR_ip_b))+ ' dB');
50 disp('Output SNR for (b) is '+string(10*log10(
    SNR_op_b))+ ' dB');
51
52 // Part (c)
53
54 //Input Signal energy Si = 0.01
55 Si_c = 0.01;
56
57 //No of levels used for PCM Coding M = 256
58 M_c = 256;
59
60 N_c = log2(M_c);
61
62 //Input SNR is SNR_ip_c
63 SNR_ip_c = Si_c/(n*fM);
64
65 //Output SNR is SNR_op_c
66 SNR_op_c = (2^(2*N_c))/(1 + (2^(2*N_c + 1))*erfc((
    SNR_ip_c*(1/(2*N_c))))^0.5);
67
68 disp('Input SNR for (c) is '+string(10*log10(
    SNR_ip_c))+ ' dB');
69 disp('Output SNR for (c) is '+string(10*log10(
    SNR_op_c))+ ' dB');

```

---

### Scilab code Exa 12.3 SNR of Optimal reciver

```

1 clc;
2 //page 609
3 //problem 12.3
4
5 //Baseband cutoff signal fM = 4 kHz

```

```

6 fM = 4 * 10^3;
7
8 //White noise power spectral density n
9 n = 2*10^(-9);
10
11 // Part (a)
12
13 //Input Signal energy Si = 0.001
14 Si = 0.001;
15
16 //No of levels used for PCM Coding M = 8
17 M = 8;
18
19 N = log2(M);
20
21 //Input SNR is SNR_ip
22 SNR_ip = Si/(n*fM);
23
24 //Output SNR is SNR_op
25 SNR_op = (2^(2*N))/(1 + (2^(2*N + 1))*erfc((SNR_ip
        *(3/(10*N))))^0.5);
26
27 disp('Input SNR for (a) is '+string(10*log10(SNR_ip)
        )+' dB');
28 disp('Output SNR (a) is '+string(10*log10(SNR_op))+
        ' dB');
29
30 // Part (b)
31
32 //Input Signal energy Si = 0.001
33 Si = 0.001;
34
35 //No of levels used for PCM Coding M = 256
36 M_b = 256;
37
38 N_b = log2(M_b);
39
40 //Input SNR is SNR_ip_b

```

```

41 SNR_ip_b = Si/(n*fM);
42
43 //Output SNR is SNR_op_b
44 SNR_op_b = (2^(2*N_b))/(1 + (2^(2*N_b + 1))*erfc((
    SNR_ip_b*(3/(10*N_b))))^0.5);
45
46 //Unfortunately in scilab the function erfc
    approximates the output value to a larger extent
    due to which an exact value cannot be obtained.
47 //The difference in the textbook answer and obtained
    answer is significant because of converting the
    answer into dB.
48
49 disp('Input SNR for (b) is '+string(10*log10(
    SNR_ip_b))+ ' dB');
50 disp('Output SNR for (b) is '+string(10*log10(
    SNR_op_b))+ ' dB');
51
52 // Part (c)
53
54 //Input Signal energy Si = 0.01
55 Si = 0.01;
56
57 //No of levels used for PCM Coding M = 256
58 M = 256;
59
60 N = log2(M);
61
62 //Input SNR is SNR_ip_c
63 SNR_ip_c = Si/(n*fM);
64
65 //Output SNR is SNR_op_c
66 SNR_op_c = (2^(2*N))/(1 + (2^(2*N + 1))*erfc((
    SNR_ip_c*(3/(10*N))))^0.5);
67
68 disp('Input SNR for (c) is '+string(10*log10(
    SNR_ip_c))+ ' dB');
69 disp('Output SNR for (c) is '+string(10*log10(

```

```
SNR_op_c))+' dB');
```

---

### Scilab code Exa 12.4 Output SNR in DM including Thermal Noise

```
1  clc;
2  //page 618
3  //problem 12.4
4
5  //Upper cut off frequency fb = 3200 Hz
6  fM = 3200;
7
8  //Lower cut off frequency fl = 300 Hz
9  fl = 300;
10
11 //Data rate fb = 32000 bps
12 fb = 32000;
13
14 //White noise power spectral density n
15 n = 2*10^(-9);
16
17 //Input Signal energy Si = 0.001
18 Si = 0.001;
19
20 //Output SNR is SNR_op
21 SNR_op = (0.6*(fb/fM)^3)/(1 + (0.3*(fb^2/(fl*fM)))*
           erfc(Si/(n*fb)));
22
23 disp('Output SNR is '+string(10*log10(SNR_op))+ 'dB'
       ');
24
25 //Data rate fb_n = 32000 bps
26 fb_n = 2*32000;
27
28 //Output SNR is SNR_op_n
29 SNR_op_n = (0.6*(fb_n/fM)^3)/(1 + (0.3*(fb_n^2/(fl*
```



```
    fM)))*erfc(Si/(n*fb_n));  
30  
31 disp('Output SNR when data rate is doubled is '+  
    string(10*log10(SNR_op_n))+ 'dB');
```

---

# Chapter 13

## Information Theory and Coding

Scilab code Exa 13.1 Information rate of source

```
1  clc;
2  //page 631
3  //problem 13.1
4
5  //Given probabilities p1 = p4 = 1/8 & p2 = p3 = 3/8
6  p1 = 1/8
7  p4 = 1/8
8  p2 = 3/8
9  p3 = 3/8
10
11 //The average information H is p1*log2 (1/p1)+p2*
    log2 (1/p2)+p3*log2 (1/p3)+p4*log2 (1/p4) bits/
    message
12 H = p1*log2 (1/p1)+p2*log2 (1/p2)+p3*log2 (1/p3)+p4*
    log2 (1/p4)
13
14 //information rate R is r*H bits/sec where r is 2*B
15 //R1 = R/B
16 R1 = 2*H
```

```

17
18 disp('The average information is '+string(H)+' bits/
    message ')
19 disp('Information rate '+string(R1)+'*B bits/sec ')

```

---

### Scilab code Exa 13.4 Channel Capacity

```

1  clc;
2  //page 649
3  //problem 13.4
4
5  //Given bandwidth(B) = 4000Hertz & Noise PSD(n/2) =
    10^-9 Watt/Hertz
6  B = 4000
7  n = 2*10^-9
8
9  //Chanel capacity(C) = B*log2 (1+S/(n*B))
10
11 //case 1
12 //Signal energy(S) = 0.1 Joule
13 S = 0.1
14
15 C = B*log2 (1+S/(n*B))
16
17 disp('Channel capacity for bandwidth = 4000Hertz ,
    Noise PSD = 10^-9 Watt/Hertz & Signal energy(S) =
    0.1 Joule is '+string(C)+' bits/sec ')
18
19 //case 2
20 //Signal energy(S) = 0.001 Joule
21 S = 0.001
22
23 C = B*log2 (1+S/(n*B))
24
25 disp('Channel capacity for bandwidth = 4000Hertz ,

```

```

    Noise PSD = 10^-9 Watt/Hertz & Signal energy (S) =
        0.001 Joule is '+string(C)+' bits/sec')
26
27 //case 3
28 //Signal energy (S) = 0.001 Joule & increased
    bandwidth (B) = 10000 Hertz
29 B = 10000
30 S = 0.001
31
32 C = B*log2 (1+S/(n*B))
33
34 disp('Channel capacity for bandwidth = 10000 Hertz ,
    Noise PSD = 10^-9 Watt/Hertz & Signal energy (S) =
        0.001 Joule is '+string(C)+' bits/sec')

```

---

### Scilab code Exa 13.8 Probability of error

```

1  clc;
2  //page 675
3  //problem 13.8
4
5  //With single parity bit added, the code size = 4.
    An error evades parity check if any 2 or all
    symbols of the code arrives are erroneous.
6  //Probability of any symbol from n are erroneous =
    nCm*(p^m)*(1-p)^(n-m)
7
8  //Thus, the probability of error undetected,
    P_undeterr = (4C2*(p^2)*(1-p)^2)+4C4*(p^4) = 6*(p
    ^2)*(1-p)^2+(p^4)
9
10 //Probability of error in detection (P_undeterr1) for
    p = 0.1
11 p = 0.1
12 P_undeterr1 = 6*(p^2)*((1-p)^2)+(p^4)

```

```

13
14 disp('Probability of error in detection for p = 0.1
      is '+string(P_undeterr1))
15
16 //Probability of error in detection(P_undeterr2) for
      p = 0.01
17 p = 0.01
18 P_undeterr2 = 6*(p^2)*((1-p)^2)+(p^4)
19
20 disp('Probability of error in detection for p = 0.01
      is '+string(P_undeterr2))

```

---

#### Scilab code Exa 13.11 Turbo code

```

1  clc;
2  //page 696
3  //problem 13.11
4
5  //The output equations are as follows v1 = s1 xor s2
      xor s3 & v2 = s1 xor s3
6  //the no of bits in output mode(bits_out) is v*(L+K)
      , v = no of outputs for commutatot = 2, L =
      length of input = 3 & K = no of memeory elements
      = 3
7  v = 2
8  L = 3
9  K = 3
10 bits_out = v*(L+K)
11
12 //Taking in , s1 , s2 , s3 , v1 & v2 as row matrix
      where each column represents its corresponding
      input or output , in means input
13 in = [0 1 0 1 0 0 0]
14
15 s1 = zeros(1,7)

```

```

16 s2 = zeros(1,7)
17 s3 = zeros(1,7)
18 v1 = zeros(1,7)
19 v2 = zeros(1,7)
20
21
22 for i = 2:7
23     s3(i) = s2(i-1)
24     s2(i) = s1(i-1)
25     s1(i) = in(i-1)
26     v1(i-1) = bitxor(s1(i),bitxor(s2(i),s3(i)))
27     v2(i-1) = bitxor(s1(i),s3(i))
28 end
29
30 //Output matrix is out
31 out = zeros(1,12)
32 for i = [1 3 5 7 9 11]
33     out(i) = v1((i+3)/2)
34     out(i+1) = v2((i+3)/2)
35 end
36
37 disp('output is ')
38 disp(out)

```

---

### Scilab code Exa 13.12 Turbo code

```

1  clc;
2  //page 697
3  //problem 13.12
4
5  //The generatr matrix requires impulse response of
   the coder.
6  //This is the ourput generated when the initially
   reset coder is fed with a single 1.
7  //The no of bits in the output code =  $2(1+3) = 8$ 

```

```

8
9 //Taking in , s1 , s2 , s3 , v1 & v2 as row matrix
   where each column represents its corresponding
   input or output , in means input
10 in = [0 1 0 0 0]
11
12 s1 = zeros(1,5)
13 s2 = zeros(1,5)
14 s3 = zeros(1,5)
15 v1 = zeros(1,5)
16 v2 = zeros(1,5)
17
18
19 for i = 2:5
20     s3(i) = s2(i-1)
21     s2(i) = s1(i-1)
22     s1(i) = in(i-1)
23     v1(i-1) = bitxor(s1(i),bitxor(s2(i),s3(i)))
24     v2(i-1) = bitxor(s1(i),s3(i))
25 end
26
27 //Output matrix is out
28 out = zeros(1,8)
29 for i = [1 3 5 7]
30     out(i) = v1((i+3)/2)
31     out(i+1) = v2((i+3)/2)
32 end
33
34 disp('impulse response is')
35 disp(out)
36
37 //Then generator matrix is G
38 G = [1 1 1 0 1 1 0 0 0 0 0 0;0 0 1 1 1 0 1 1 0 0 0
      0;0 0 0 0 1 1 1 0 1 1 0 0]
39
40 //Note that , in G, impulse responses appear in
   staggered apper in a staggered manner in each row
   while the rest of the elements are 0.

```

```

41
42 //Now, output is b_o = b_i*G where input b_i =[1 0
    1]
43 b_i = [1 0 1]
44
45 b_o = b_i*G
46
47 //Here multiplication means Exor operation so
    wherever two occurs it should be changed to 1
48
49 for i = 1:12
50     if b_o(i) > 1 then
51         b_o(i) = 0;
52     end
53 end
54
55 disp('output is ')
56 disp(b_o)
57 disp('The output obtained is exactly same as example
    13.1 ')

```

---

#### Scilab code Exa 13.14 GO back N algorithm

```

1  clc;
2  //page 701
3  //problem 13.14
4
5  //Given, Tw = 10microsec, BCH(1023973) code is used
    implies k as 973 & n as 1023, P_A = 0.99, T1 = 40
    microsec & N = 4
6  Tw = 10*10^-6
7  k = 973
8  n = 1023
9  P_A = 0.99
10 T1 = 40*10^-6

```



```

11 N = 4
12
13 // efficiency of Stop-and-Wait ARQ(n_SandW) = (k/n)*(
    P_A/(1+(T1/Tw)))
14 n_SandW = (k/n)*(P_A/(1+(T1/Tw)))
15
16 // efficiency of Go-Back-N ARQ(n_GBN) = (k/n)*(1/(1+(
    N*(1-P_A)/P_A)))
17 n_GBN = (k/n)*(1/(1+(N*(1-P_A)/P_A)))
18
19 // efficiency of Selective Repeat ARQ(n_SR) = (k/n)*
    P_A
20 n_SR = (k/n)*P_A
21
22 disp('efficiency of Stop-and-Wait ARQ is '+string(
    n_SandW))
23 disp('efficiency of Go-Back-N ARQ is '+string(n_GBN)
    )
24 disp('efficiency of Selective Repeat ARQ is '+string
    (n_SR))

```

---

### Scilab code Exa 13.15 Power of a Transmitter

```

1  clc;
2  //page 718
3  //problem 13.15
4
5  //Bit interval T = 1/10^6 = 10^-6 sec
6  T = 10^-6
7
8  //White Noise Power Spectral Density n/2 = 10^-9 W/
    Hz
9  n = 2*10^-9
10
11 //Power required Ps = Eb/T, where Eb = energy per

```

```

        bit
12
13 //For information system feedback system  $E_b = n$ 
14  $P_s = n/T$ 
15
16 disp('power required for information system feedback
        system is '+string(Ps)+' Watt')
17
18 //For optimal system  $P_s = (0.69 * n)/T$ 
19  $P_s = (0.69 * n)/T$ ;
20
21 disp('power required for optimal system is '+string(
        Ps)+' Watt')
```

---

**Scilab code Exa 13.16** Probability of error for Trellis decoded modulation

```

1  clc;
2  //page 719
3  //problem 13.16
4
5  //Given,  $E_b = 10^{-9}$  Joule,  $n/2 = 10^{-9}$  Watt/Hertz
6   $E_b = 10^{-8}$ 
7   $n = 2*10^{-9}$ 
8
9  //Probability of error for trellis-decoded
        modulation( $P_e$ ) = (1/2)*erfc(sqrt(1.5* $E_b/n$ ))
10  $P_e = (1/2)*erfc(sqrt(1.5*E_b/n))$ 
11
12 disp('Probability of error for trellis-decoded
        modulation is '+string( $P_e$ ))
13
14 //Probability of error for Qpsk modulation( $P_e$ ) =
        (1/2)*erfc(sqrt( $E_b/n$ ))
15  $P_e = (1/2)*erfc(sqrt(E_b/n))$ 
16
```

```
17 disp('Probability of error for Qpsk modulation is '+  
    string(Pe))
```

---

# Chapter 14

## Communication Systems and Component Noises

Scilab code Exa 14.1 Thermal noise voltage

```
1  clc;
2  //page 738
3  //problem 14.1
4
5  //Boltzman constant k = 1.3806488      10-23 m2 kg s
      -2 K-1
6  k = 1.3806488 * 10-23;
7
8  //Let room temperature be 27 C
9  T = 27 + 273;
10
11 //Bandwidth BW = 10 MHz
12 BW = 10 * 106;
13
14 //For (a)
15 //Let the equivalent resistance be Ra
16 Ra = 10 + 10;
17
18 //RMS Noise Voltage be Va
```

```

19 Va = (4*k*T*Ra*BW)^0.5;
20
21 disp('The rms voltage at output a is '+string(Va)+'
      Volt');
22
23 //For (b)
24 //Let the equivalent resistance be Rb
25 Rb = (10 * 10)/(10+10);
26
27 //RMS Noise Voltage be Vb
28 Vb = (4*k*T*Rb*BW)^0.5;
29
30 disp('The rms voltage at output b is '+string(Vb)+'
      Volt');
31
32 //For (c)
33
34 Rc = 10;
35 C = 1*10^-9;
36
37 //In the textbook, the author has forgotten to
      multiply the result with T, hence has obtained an
      erroneous result.
38 //The given answer is 28.01uV but the correct answer
      is found out to be 1.2uV
39
40 Vc_square = 2*k*integrate('Rc/(1 + (2*%pi*Rc*C*f)^2)
      ','f',-10^7,10^7);
41 Vc = Vc_square^0.5;
42
43 disp('The rms voltage at output c is '+string(Vc)+'
      Volt');

```

---

Scilab code Exa 14.2 Output Noise power

```

1  clc;
2  //page 741
3  //problem 14.2
4
5  //The Antenna noise temperature is T_ant = 10 K
6  T_ant = 10;
7
8  //The reciever noise temperature is Te = 140 K
9  Te = 140;
10
11 //Midband available gain of reciever gao = 10^10
12 gao = 10^10;
13
14 //Noise bandwidth is BN = 1.5 * 10^5 Hz
15 BN = 1.5 * 10^5;
16
17 //Boltzman constant k = 1.3806488      10-23 m2 kg s
      -2 K-1
18 k = 1.3806488 * 10-23;
19
20 //Available noise power at output is pao
21
22 pao = gao*k*(T_ant + Te)*BN;
23
24 disp('The available output noise power is '+string(
      pao)+' Watts');
```

---

### Scilab code Exa 14.3 Transmitted power of antenna

```

1  clc;
2  //page 748
3  //problem 14.3
4
5  //The distance d = 30 * 1.6 * 10^3 m;
6  d = 30 * 1.6 * 10^3;
```

```
7
8 //Frequency f = 4 * 10^9 Hz
9 f = 4 * 10^9;
10
11 //Wavelength w = c/f m
12 w = 3*10^8 / f;
13
14 //Transmitter gain KT = 40 dB
15 KT = 10^4;
16
17 //Reciever gain KR = 40 dB
18 KR = 10^4;
19
20 //Reciever power PR = 10^-6 Watt
21 PR = 10^-6;
22
23 //Transmitter power PT
24 PT = PR*(4*%pi*d/w)^2/ (KT*KR);
25
26 disp('The transmitter output is '+string(PT)+' Watt'
      )
```

---

# Chapter 15

## Spread Spectrum Modulation

Scilab code Exa 15.1 Jamming

```
1  clc;  
2  //page 764  
3  //problem 15.1  
4  
5  //Signal Power data rate fb = 100 Kbps  
6  fb = 105;  
7  
8  //Signal Strength Ps = 1 mW  
9  Ps = 1*10(-3);  
10  
11 //Chip frequency fs = 100 MHz  
12 fs = 108;  
13  
14 //Noise Spectral Density n = 2*10(-9) W/Hz  
15 n = 2*10(-9);  
16  
17 //Jamming Signal power is Pj = 1 W  
18 Pj = 1;  
19  
20 //Processing Gain P  
21 P = fs/fb;
```



```

22 disp('Processing Gain is '+string(P));
23
24 //Bit Interval T
25 T = 1/fb;
26 disp('Bit Interval is '+string(T)+'s');
27
28 //Energy per bit Eb
29 Eb = Ps*T;
30 disp('Energy per bit is '+string(Eb));
31
32 //Error Probability without jamming
   E_without_jamming
33 E_without_jamming = 0.5*erfc((Eb/(n))^0.5);
34 disp('Error probability without jamming is '+string(
   E_without_jamming));
35
36 //Error Probability with jamming E_jamming
37 E_jamming = 0.5*erfc(((2*Ps*P)/(Pj))^0.5);
38 disp('Error probability jamming is '+string(
   E_jamming));

```

---

### Scilab code Exa 15.2 Ranging using DS spread spectrum

```

1  clc;
2  //page 764
3  //problem 15.2
4
5  //Chip Rate fc = 110 MHz
6  fc = 10*10^6;
7  Tc = 1/fc;
8
9  //Delay D = 0.1 ms
10 D = 0.1*10^-3;
11
12 //Speed of light c = 3*10^8 Km/s

```

```

13 c = 3*10^8;
14
15 //Estimated Distance d
16 d = 0.5*c*D;
17
18 //Tolerance Tol
19 Tol = 0.5*c*Tc;
20
21 disp('The target is between '+string(d-Tol)+' metres
      and '+string(d+Tol)+' metres of the source.');
```

---

#### Scilab code Exa 15.3 Sequence length

```

1  clc;
2  //page 769
3  //problem 15.3
4
5  //Number of Flip Flops N
6  N = 13;
7
8  //Maximal length of sequence L
9  L = 2^N - 1;
10
11 //Upper Bound S
12 S = (L - 1)/N;
13
14 //No of basic sequences and mirror images
15 disp('No of basic sequences and mirror images is '+
      string(S/2)');
```

---