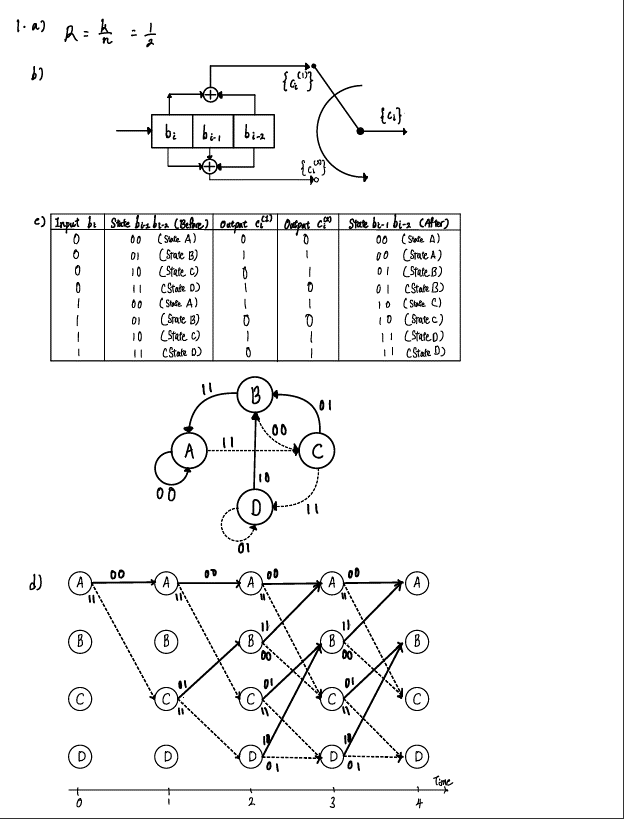
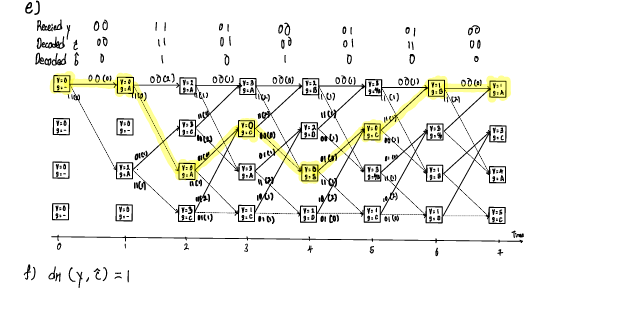
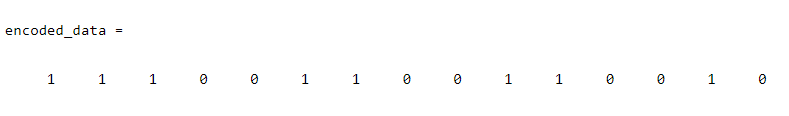
*陳志偉 B06901126電機五*

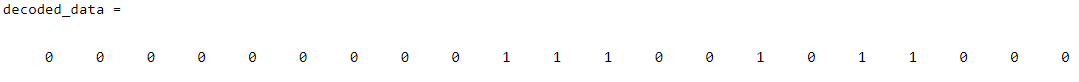
**Electrical Engineering Lab（topics on Communication System）**

**Lab5 Report**

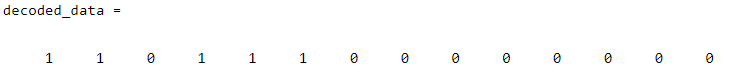


2.a)

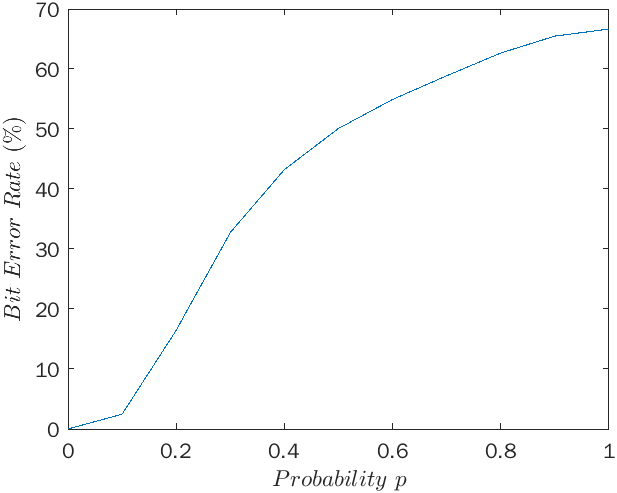
2.b)

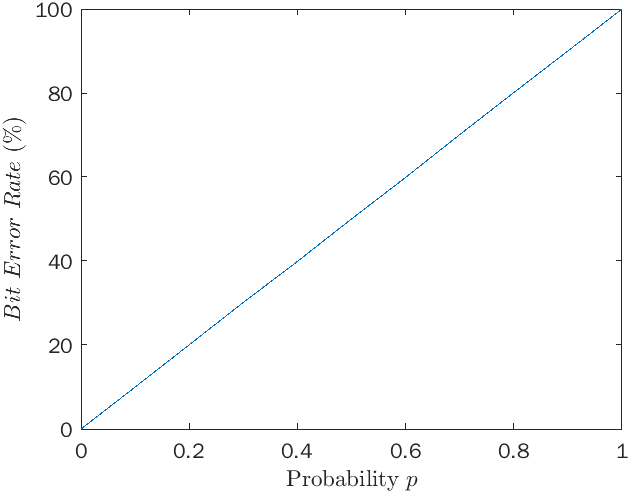


Verity Problem 1e.



2.c) When p > ½, the slope of *Bit Error Rate* and *Probability* will become lower.



3) By comparing with 2.c), the slope of *Bit Error Rate* and *Probability* will always equal to 1.

***Appendix***

**Code of Lab 5**

1. %2.a)
2. ge = [1, 0, 0; 1, 0, 1; 1, 1, 1];
3. binary\_data = [1, 0, 1, 1, 0];
4. encoded\_data = conv\_enc(binary\_data, ge);
5. display(encoded\_data)
6. %2.b)
7. d = [0 1 0 0 0 0 1 0 1 1 1 1 0 0 1 0 1 1 0 0 0];
8. decoded\_data = conv\_dec(d, ge);
9. display(decoded\_data)
10. y = [0 0 1 1 0 1 0 0 0 1 0 1 0 0];
11. ge = [1, 0, 1; 1, 1, 1];
12. decoded\_data = conv\_dec(y, ge);
13. display(decoded\_data)
14. % 2.c)
15. ber = [];
16. %ge = [1, 0, 0; 1, 0, 1; 1, 1, 1];
17. ge = [1, 1, 0; 1 0 1];
18. for p = 0:0.1:1
19. x = randi([0 1],110000,1);
20. encoded\_data = conv\_enc(x, ge);
21. noised\_data = bsc(encoded\_data, p);
22. decoded\_data = conv\_dec(noised\_data, ge);
23. bit\_error\_rate = 0;
24. for n = 1:length(decoded\_data)
25. if encoded\_data(n) ~= decoded\_data(n)
26. bit\_error\_rate = bit\_error\_rate + 1;
27. end
28. end
29. ber = [ber, bit\_error\_rate/length(encoded\_data)\*100];
30. end
31. x = linspace(0, 1, 11);
32. plot(x, ber)
33. xlabel('Probability $p$','Interpreter','latex')
34. ylabel('$Bit$ $Error$ $Rate$ $\left(\%\right)$','Interpreter','latex')
35. function coded\_bits = convention(bits, generators)
36. for n = 1:length(generators(:,1))
37. if sum(generators(n,:)) == 1
38. coded\_bits(n) = bits(1);
39. elseif sum(generators(n,:)) == 2
40. coded\_bits(n) = xor(bits(1), bits(3));
41. elseif sum(generators(n,:)) == 3
42. coded\_bits(n) = xor(xor(bits(1), bits(2)), bits(3));
43. end
44. end
45. end
46. function encoded\_data = conv\_enc(binary\_data, impulse\_response)
47. s = [0 0; 0 1; 1 0; 1 1];
48. encoded\_data = [];
49. current\_s = s(1,:);
50. for n = 1:length(binary\_data)
51. bits = [binary\_data(n), current\_s];
52. c = convention(bits, impulse\_response);
53. encoded\_data = [encoded\_data, c];
54. current\_s = [binary\_data(n), current\_s(1)];
55. end
56. end
57. function decoded\_data = conv\_dec(binary\_data, impulse\_response)
58. s = [0 0; 0 1; 1 0; 1 1];
59. row = length(impulse\_response(:,1));
60. col = length(impulse\_response(1,:));
61. decoded\_data = [];
62. v = zeros(4, length(binary\_data)/row);
63. s\_from = zeros(4, length(binary\_data)/row);
64. %Start
65. subdata = binary\_data(1:row);
66. c0 = convention([0,s(1,:)], impulse\_response);
67. c1 = convention([1,s(1,:)], impulse\_response);
68. d0 = 0;
69. d1 = 0;
70. for i = 1:length(subdata)
71. d0 = d0 + xor(subdata(i), c0(i));
72. d1 = d1 + xor(subdata(i), c1(i));
73. end
74. ste\_idx = find((s(:, 1) == 0 & s(:, 2) == s(1, 1)) == 1);
75. v(ste\_idx, 1) = d0;
76. s\_from(ste\_idx, 1) = 1;
77. ste\_idx = find((s(:, 1) == 1 & s(:, 2) == s(1, 1)) == 1);
78. v(ste\_idx, 1) = d1;
79. s\_from(ste\_idx, 1) = 1;
80. for n = 1:length(v)-1
81. subdata = binary\_data(row\*n+1:row\*n+row);
82. for k = 1:length(s(:,1))
83. if v(k, n) ~= 0
84. c0 = convention([0, s(k,:)], impulse\_response);
85. c1 = convention([1, s(k,:)], impulse\_response);
86. d0 = 0;
87. d1 = 0;
88. for i = 1:length(subdata)
89. d0 = d0 + xor(subdata(i), c0(i));
90. d1 = d1 + xor(subdata(i), c1(i));
91. end
92. ste\_idx = find((s(:, 1) == 0 & s(:, 2) == s(k, 1)) == 1);
93. if v(ste\_idx, n+1) == 0
94. v(ste\_idx, n+1) = v(k, n) + d0;
95. s\_from(ste\_idx, n+1) = k;
96. else
97. if v(ste\_idx, n+1) > v(k, n) + d0
98. v(ste\_idx, n+1) = v(k, n) + d0;
99. s\_from(ste\_idx, n+1) = k;
100. end
101. end
102. ste\_idx = find((s(:, 1) == 1 & s(:, 2) == s(k, 1)) == 1);
103. if v(ste\_idx, n+1) == 0
104. v(ste\_idx, n+1) = v(k, n) + d1;
105. s\_from(ste\_idx, n+1) = k;
106. else
107. if v(ste\_idx, n+1) > v(k, n) + d1
108. v(ste\_idx, n+1) = v(k, n) + d1;
109. s\_from(ste\_idx, n+1) = k;
110. end
111. end
112. end
113. end
114. end
115. s\_back = 0;
116. s\_now = 0;
117. for n = length(v):-1:1
118. if n == length(v)
119. s\_back = s\_from(1, n);
120. s\_now = 1;
121. else
122. s\_now = s\_back;
123. s\_back = s\_from(s\_back, n);
124. end
125. c0 = convention([0, s(s\_back,:)], impulse\_response);
126. c1 = convention([1, s(s\_back,:)], impulse\_response);
127. if [0, s(s\_back,1)] == s(s\_now,:)
128. decoded\_data = [c0, decoded\_data];
129. elseif [1, s(s\_back,1)] == s(s\_now,:)
130. decoded\_data = [c1, decoded\_data];
131. end
132. end
133. end

*All code source will push to my github repo:* [*https://github.com/finalwee/CommLab*](https://github.com/finalwee/CommLab)