15ECE401 - Information theory and Coding Techniques - Mini Project

Topic - “Determining the Various Entropies and Mutual Information of a Binary Symmetric Channel”

P. H. Lakshmi - BL.EN.U4ECE17105
P. Vignesh - BL.EN.U4ECE17144
Palli Ashish - BL.EN.U4ECE17149

Aim:

To determine various entropies and mutual information of A Binary Symmetric Channel (BSC)

Theory:

In information theory, the **entropy** of a random variable is the average level of "**information**", "**surprise**", or "**uncertainty**" inherent in the variable's possible outcomes.

Binary Symmetric Channel - BSC

- **A BSC channel** is characterized by **-**
  No.of input = No. of output = 2
- This is the most commonly and widely used channels.
- Two signals with probabilities **P(A)** and **P(B). These probabilities together must add up to 1**.

P(A) = w and P(B) = $\overline{w}$, such that $w + \overline{w} = \mathbf{1}$

- **Conditional Probability Matrix** also called the **Channel Matrix,** is as follows -

$$P(Y/X) = \begin{bmatrix} \bar{p} & p \\ p & \bar{p} \end{bmatrix}$$

Where p is the **Probability of Error.** This is the probability of that event **signal 1 is passed** but **signal 2 is received** at the receiver, and vice versa. $\bar{p}$ is the probability when the **correct signal is received** at the receiver.

- **Equivocation (h)** is given by

$$H(Y/X) = h = \sum_{j=1}^{2} p_j \log \frac{1}{p_j}$$

$$\therefore H(Y/X) = h = \bar{p} \log \frac{1}{\bar{p}} + p \log \frac{1}{p}$$

- **Entropy of Output Symbols H(Y) -** given by the formula

$$H(Y) = \sum_{j=1}^{2} P(y_j) \log \frac{1}{P(y_j)}$$

$$= P(y_1) \log \frac{1}{P(y_1)} + P(y_2) \log \frac{1}{P(y_2)}$$

In order to calculate $H(y)$, we need the values $P(y1)$ and $P(y2)$. Here $P(Y1)$ and $P(Y2)$ are calculated as follows -

$$\boxed{\begin{aligned} & P(y_1) = P(y_1/x_1)\,P(x_1) + P(y_1/x_2)\,P(x_2) = \bar{p}\,w + p\,\bar{w} \\ \text{and}\quad & P(y_2) = P(y_2/x_1)\,P(x_1) + P(y_2/x_2)\,P(x_2) = pw + \bar{p}\,\bar{w} \end{aligned}}$$

After which, $H(y)$ is calculated as follows -

$$\boxed{H(Y) = (\bar{p}\,w + p\,\bar{w})\log\frac{1}{(\bar{p}w + p\bar{w})} + (pw + \bar{p}\,\bar{w})\log\frac{1}{(pw + \bar{p}\bar{w})}}$$

➢ **Mutual Information** $I(x, y)$ is given by the following formula -

$$\boxed{I(Y, X) = I(X, Y) = H(Y) - H(Y/X)}$$

➢ **Channel Capacity [C] -**

$$\boxed{\begin{aligned} C &= \log s - h \text{ bits/sec} \\ &= \log 2 - h = 1 - h \end{aligned}}$$

➢ **Rate of Transmission (R_t)**

$$\boxed{R_t = I(X, Y)\, r_s}$$

➢ **Source Efficiency** ($\eta_{channel}$) and **Redundancy** ($R_{\eta\text{-}channel}$) -

Source Efficiency ($\eta_{channel}$) = H(X) / $H(X)_{max}$
Source Redundancy ($R_{\eta\text{-}channel}$) = 1 - ($\eta_{channel}$)

Algorithm:

1. The total number of inputs and outputs to and from a Binary Symmetric Channel (BSC) is fixed, i.e, 2.
2. Input one input probability from the user. The Second probability is calculated automatically since the sum of probabilities must be equal to 1. These probabilities are denoted by P(A) and P(B). These are also denoted as w and $\overline{w}$.
3. Input the probability of error (p)
4. The Channel matrix is computed using the probability of error. It is denoted by P(Y/X).
5. Calculate the entropy of the channel input. i.e. H(X)
6. Calculate output probabilities P(Y1) and P(Y2) using p and w computed in steps 2 and 3. Also, calculate the entropy of channel output. i.e. H(Y).
7. Compute Equivocation H(X/Y)
8. Compute Mutual Information I(X, Y) using **I(Y, X)=H(Y)-H(Y/X)**
9. Compute Joint Entropy H(X, Y)
10. Compute Transmission Rate, Channel Capacity, Source Efficiency and Redundancy.

Conclusion:

In this project, we compute the various Entropies, Mutual Information, Efficiency and Redundancy of signals passed through a Binary Symmetric Channel (BSC). We observe that for equiprobable signals -

a. Entropy of the output is unity.
b. Mutual Information maximises and becomes equal to the Channel Capacity [C].

Information Theory and Coding Techniques - Mini Project

Calculation of Entropies and Mutual Information using a Binary Symmetric Channel

---

> **Group**
>
> - **P. H. LAKSHMI** - BL.EN.U4ECE17105
> - **P. VIGNESH** - BL.EN.U4ECE17144
> - **PALLI ASHISH** - BL.EN.U4ECE17149

- **math library** - this library is used to implement a predefined function - log()
- **numpy library** - this library is used to implement a predefined function - reciprocal()

```
#Loading the necessary libraries
import math
import numpy as np
```

- The **Number of Inputs** and **Outputs** for a BSC is 2.
- **Probabilities of symbols** taken as user input.
- **Probability of error** is also taken as a user input.

```
#Number of inputs and outputs to and from a Binary Symmetric Channel is always 2
inputs = 2
outputs = 2


#User input of probabilities of transmitted symbols
prob_a = float(input("probability of symbol 1 : "))

#since sum of probabilites must be equal to 1
prob_b = 1 - prob_a

#User Input of the Probability of Error
p = float(input("Enter the probability of error (p) (bits/message-symbol): "))
p_bar = 1-p
```

```
probability of symbol 1 : 0.4
Enter the probability of error (p) (bits/message-symbol): 0.04
```

Input - Channel Matrix - P(Y/X)

```
#Channel matrix initialised as an array (array of arrays)
CM = [[p_bar,p],
      [p,p_bar]]
print(CM)
```

```
[[0.96, 0.04], [0.04, 0.96]]
```

Channel probabilities - P(x)

```
w = prob_a
w_bar = prob_b
print(w,w_bar)
w_mat = [w,w_bar]
```

```
0.4 0.6
```

Entropy of channel input H(x)

```
H_x = 0
for i in range (0, outputs):
    var = w_mat[i]
    print(var)
    var_log = math.log((np.reciprocal(var)),2)

    H_x = H_x + var*var_log

print(H_x,"bits/message-symbol")
```

```
0.4
0.6
0.9709505944546687 bits/message-symbol
```

Entropy of channel output H(y)

```
p_y1 = (p_bar*w)+(p*w_bar)
p_y2 = (p*w)+(p_bar*w_bar)
print(p_y1,p_y2)
```

```
0.40800000000000003 0.592
```

```
log_p_y1 = math.log(np.reciprocal(p_y1),2)
log_p_y2 = math.log(np.reciprocal(p_y2),2)
print(log_p_y1,log_p_y2)
```

```
1.2933589426905914 0.7563309190331374
```

```
H_y = (p_y1*log_p_y1)+(p_y2*log_p_y2)
print(H_y,"bits/message-symbol")
```

```
0.9754383526853786 bits/message-symbol
```

Equivocation - H(X/Y)

```
log_p = math.log(np.reciprocal(p),2)
log_p_bar = math.log(np.reciprocal(p_bar),2)
H_x_y = (p_bar*log_p_bar)+(p*log_p)
print(H_x_y,"bits/message-symbol")
```

```
0.24229218908241487 bits/message-symbol
```

Mutual Information - I(X,Y)

```
I_x_y = H_y - H_x_y
print(I_x_y,"bits/message-symbol")
```

```
0.7331461636029637 bits/message-symbol
```

Joint Entropy - H(X,Y)

```
H1_x_y = H_x + H_x_y
print(H1_x_y,"bits/message-symbol")
```

```
1.2132427835370836 bits/message-symbol
```

Transmission Rate (Rt)

```
#symbol rate (rs) is assumed to be 100 symbols/second
r_s = 100
R_t = I_x_y * r_s
print(R_t,"bits/second")
```

```
print(R_t,"bits/second")
```

```
73.31461636029637 bits/second
```

Channel Capacity (C)

```
C = 1 - H_x_y
print(C,"bits/second")
```

```
0.7577078109175851 bits/second
```

Source Efficiency (Eeta_channel)

```
Eeta_channel = H_x / math.log(2,2)
print(Eeta_channel*100,"%")
```

```
97.09505944546687 %
```

Source Redundancy (R_Eeta_channel)

```
R_Eeta_channel = 1 - Eeta_channel
print(R_Eeta_channel*100,"%")
```

```
2.904940554533131 %
```