1
Our Scheme

2

Mutual Information and Channel Capacity

https://en.wikipedia.org/wiki/Mutual$_{i}nformation$
At the channel output, an observer 's average uncertainty concerning the channel input will have some value before the reception of an output, and it will usually decrease when the output is received.
Therefore, the decrease in the observer's average uncertainty of the transmitted signal when the output is received is a measure of the average transmitted information, $I(X;Y)$.

$$I(X,Y) = H(X) - H(X|Y) \quad \text{or} \quad I(X;Y) = H(Y) - H(Y|X)$$

It means that the mutual information is a function of the source probabilities as well as of the channel transition probabilities. Mutual information is always non-negative, so we can deduce the following relation:

$$H(X|Y) \leq H(X)$$

The channel capacity C is defined as the maximum value of mutual information, which is the maximum average information per symbol that can be transmitted through the channel.

$$C = max[I(X;Y)]$$

The maximization is with respect to the source probabilities, since the transition probabilities are fixed by the channel. However, the channel capacity is a function of only the channel transition probabilities, since the maximization process eliminates the dependence on the source probabilities.

3
Error Probability

The error probability P_E of a binary symmetric channel is computed as:

$$P_E = \sum_{n=1}^{2} p(e|x_i)p(x_i)$$

where $p(e|x_i)$ is the error probability given input x_i, we have

$$P_E = qp(x_1) + qp(x_2) \implies p_E = q$$

It states that the unconditional error probability P_E is eqyal to the conditional error probability $p(y_j|x_i), i \neq j$.

P_E is a decreasing function of the energy of the received symbols. Since the symbol energy is the received power multiplied by the symbol period, it follows that if the transmitter power is fixed, the error probability can be reduced by decreasing the source rate. This can be accomplished by removing the redundancy at the source through source encoding.

4

Hamming Code

We can understand how codes can detect and correct errors from a geometric point of view. A binary codeword with a sequence of 1's and 0's has n symbols in length. The weights, namely Hamming weight $w(s_j)$ of codeword s_j is defined as the number if 1's in that codeword. We also define the number of positions in which s_i and s_j as Hamming distance $d(s_i, s_j)$ or d_{ij}.

Hamming distance can be written in terms of Hamming weight as

$$d_{ij} = w(s_i \bigoplus s_j)$$

where the symbol $\oplus$ denotes modulo-2-addition, which is binary addition without a carry.
In a broader sense, Hamming distance represents the sum of corresponding elements that differ between two vectors.

$$d(s_i, s_j) = \frac{1}{n} \sum_{n=1}^{N} s_i - s_j$$

A minimum decode can always correct as many as e errors, where e is the largest integer not to exceed

$$\frac{1}{2}(d_m - 1)$$

where d_m is the minimum distance between codewords. If d_m is odd, all received words can be assigned to a codeword. However, if d_m is even, a received word can lie halfway between two codewords. in this case, even though errors can be detected, they cannot be corrected.

5 Modulation

A desirable characteristic of any modulation scheme is the simultaneous conservation of bandwidth and power. There are two approaches to combine coding and modulation. One is Continuous Phase Modulation (CPM) with memory extend over several modulation symbols by cyclical use of a set of modulation indices. Another on is combing coding with an M-ary modulation scheme, referred to as Trellis-Coded Modulation (TCM).

6

Transmission Concept

A super block is one symbol, which contains a number of blocks. Each block contains a number of time slots. For each super block length in blocks, we calculate
1. Number of ways to organize the blocks in a super block

$$n!$$

2. Number of Bits / Symbol

$$log_2(n!)$$

3. Number of Bits / Photon

$$\frac{log_2(n!)}{n}$$

4. Number of Bits / Time Slot

$$\frac{log_2(n!)}{n} \times \frac{n}{T}$$

For example, we have [1,2,4,7], there are 4! = 24 permutation of ways to organize the blocks to generate different super blocks representing the corresponding symbols as follow:

$$[1, 2, 4, 7] \to A, [1, 2, 7, 4] \to B, [1, 4, 2, 7] \to C, [1, 4, 7, 2] \to D, [1, 7, 2, 4] \to E, [1, 7, 4, 2] \to F$$

$$[2, 1, 4, 7] \to G, [2, 1, 7, 4] \to H, [2, 4, 1, 7] \to I, [2, 4, 7, 1] \to J, [2, 7, 1, 4] \to K, [2, 7, 4, 1] \to L$$

$$[4, 1, 2, 7] \to M, [4, 1, 7, 2] \to N, [4, 2, 1, 7] \to O, [4, 2, 7, 1] \to P, [4, 7, 1, 2] \to Q, [4, 7, 2, 1] \to R$$

$$[7, 1, 2, 4] \to S, [7, 1, 4, 2] \to T, [7, 2, 1, 4] \to U, [7, 2, 4, 1] \to V, [7, 4, 1, 2] \to W, [7, 4, 2, 1] \to X$$

The information content of the super block is

$$log_2(4!) = 4.6 \quad \text{bits/symbol}$$

For each photon, it contains

$$1.15 \quad \text{bits/photon}$$

For each time slot, it has

$$0.33 bits/timeslot$$

7

Our Method

We define a block as an integral number of time bins (or time slots, or other encoding sources that are orthogonal, i.e., that can be perfectly discriminated).

We represent [1,2,4,7] into a form with polarization of photon:

1 |2 |4 |7
H H0 H000 V000000, which comprises of 14 lengths of block, equivalent to 14 time slots.

Our method uses 4 photons in 14 time slots. It means that our method has:
1. 4! = 24 ways to order them
2. 4.6 bits/symbol
3. 1.15 bits/photon
4. 0.33 bits / time slot

We can check our values form in the 5th iteration of the following table.

	Photon Number	Permutation	Bits/Symbol	Bits/Photon	Bits/Time Slots
0	1.0	1.0	0.000000	0.000000	0.000
1	2.0	2.0	1.000000	0.500000	0.333
2	3.0	6.0	2.584963	0.861654	0.369
3	4.0	24.0	4.584963	1.146241	0.327
4	5.0	120.0	6.906891	1.381378	0.266
5	6.0	720.0	9.491853	1.581976	0.206
6	7.0	5040.0	12.299208	1.757030	0.164
7	8.0	40320.0	15.299208	1.912401	0.135
8	9.0	362880.0	18.469133	2.052126	0.112
9	10.0	3628800.0	21.791061	2.179106	0.092

Figure 7.1: Table

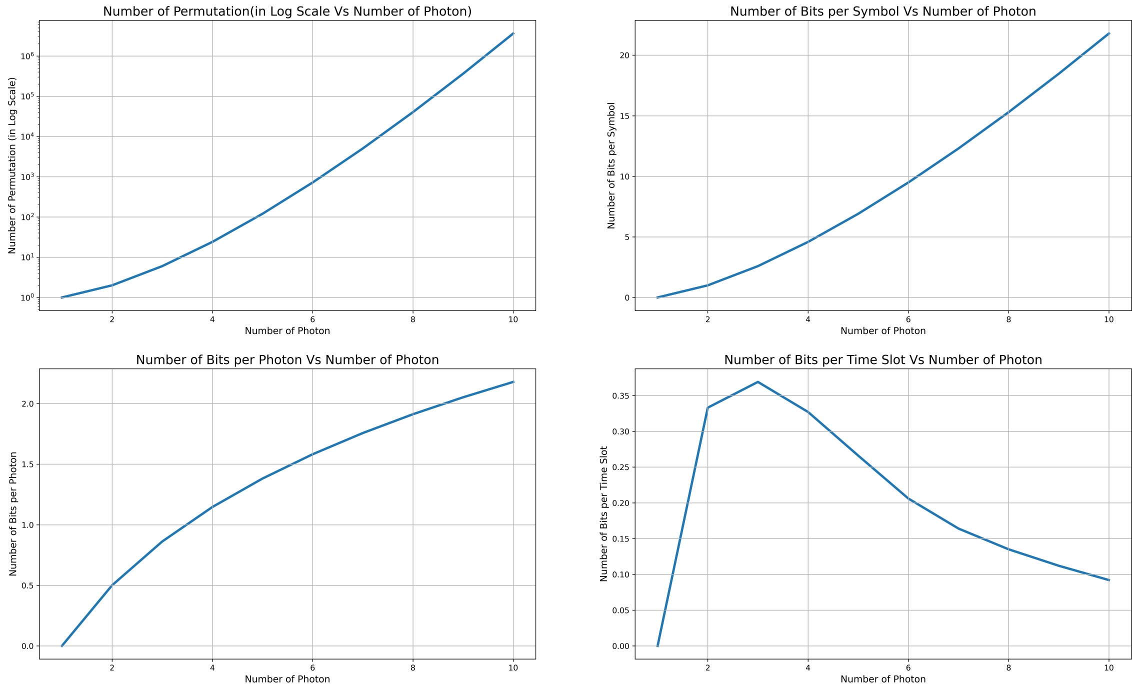

Figure 7.2: Number of Photons Vs Number of Permutation, Number of Photons Vs Number of Bits per Symbol, Number of Photons Vs Number of Bits per Photon, Number of Photons Vs Number of Bits per Time Slot

8

PPM

Pulse-Position Modulation (PPM) is a signal modulation used for both analog and digital signal transmissions, which is widely used for optical communication systems such as optical fiber and IR remote controls. In PPM, data are transmitted with short pulses. All pulses have both the same width and amplitude. The parameter that changes is the delay between each pulse.

The representation of bits to symbol is as follow:
100000 $\rightarrow A$
010000 $\rightarrow B$
001000 $\rightarrow C$
000100 $\rightarrow D$
000010 $\rightarrow E$
000001 $\rightarrow F$

PPM uses 1 photon in 14 time slots
It means that PPM has:
1. 14 ways to order them
2. 3.8 bits/symbol
3. 3.8 bits/photon
4. 0.27 bits / time slot

[?]

	Number of Photon	Number of Permutation	Number of Bits per Symbol	Number of Bits per Photon	Number of Bits per Time Slots
0	1.0	1.0	0.000000	0.000000	0.000
1	2.0	3.0	1.584963	1.584963	1.057
2	3.0	7.0	2.807355	2.807355	1.203
3	4.0	14.0	3.807355	3.807355	1.088
4	5.0	26.0	4.700440	4.700440	0.904
5	6.0	46.0	5.523562	5.523562	0.720
6	7.0	75.0	6.228819	6.228819	0.581
7	8.0	113.0	6.820179	6.820179	0.483
8	9.0	165.0	7.366322	7.366322	0.402
9	10.0	238.0	7.894818	7.894818	0.332

Figure 8.1: Table

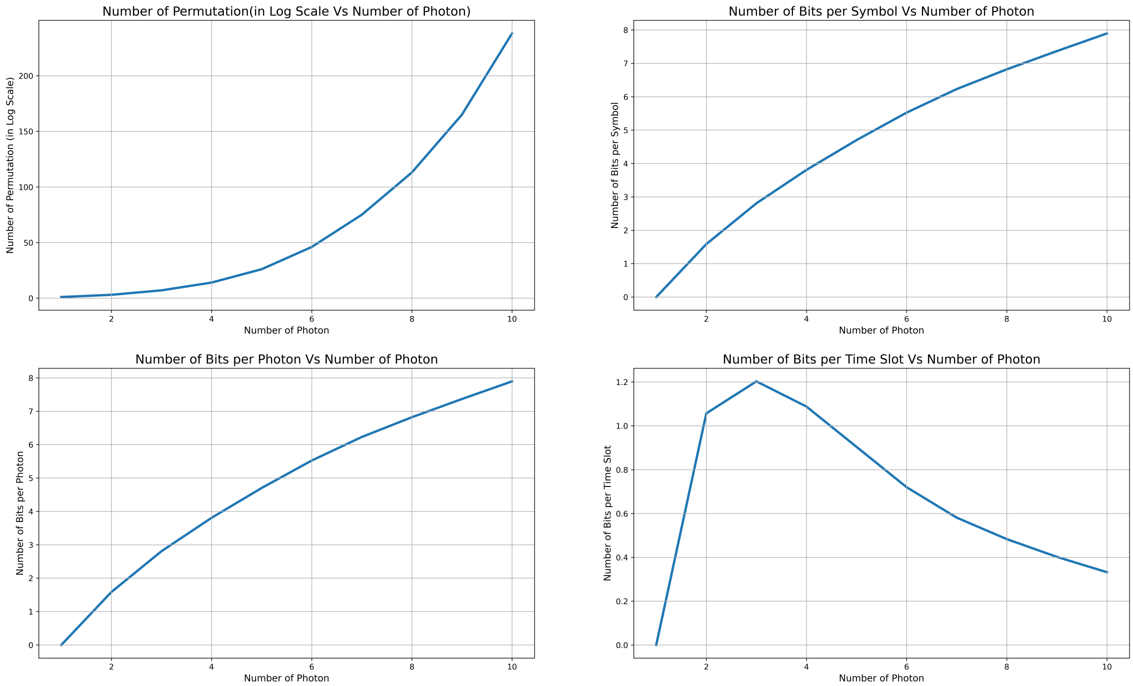

Figure 8.2: Number of Photons Vs Number of Permutation, Number of Photons Vs Number of Bits per Symbol, Number of Photons Vs Number of Bits per Photon, Number of Photons Vs Number of Bits per Time Slot

9

On-Off Key (OOK)

OOK uses 7 photons in average in 14 time slots
It means that PPM has:
1. $2^{14} = 16,384$ ways to order them
2. 14 bits/symbol
3. 2 bits/photon
4. 1 bits / time slot

	Number of Photon	Number of Permutation	Number of Bits per Symbol	Number of Bits per Photon	Number of Bits per Time Slots
0	1.0	2.0	1.0	2.0	2.000
1	2.0	8.0	3.0	2.0	1.333
2	3.0	128.0	7.0	2.0	0.857
3	4.0	16384.0	14.0	2.0	0.571
4	5.0	67108864.0	26.0	2.0	0.385
5	6.0	70368744177664.0	46.0	2.0	0.261
6	7.0	37778931862957161709568	75.0	2.0	0.187
7	8.0	10384593717069655257060992658440192	113.0	2.0	0.142
8	9.0	4676805239458889338251791464692105662898984137...	165.0	2.0	0.109
9	10.0	4417117661945960823958243751857296289568709742...	238.0	2.0	0.084

Figure 9.1: Table

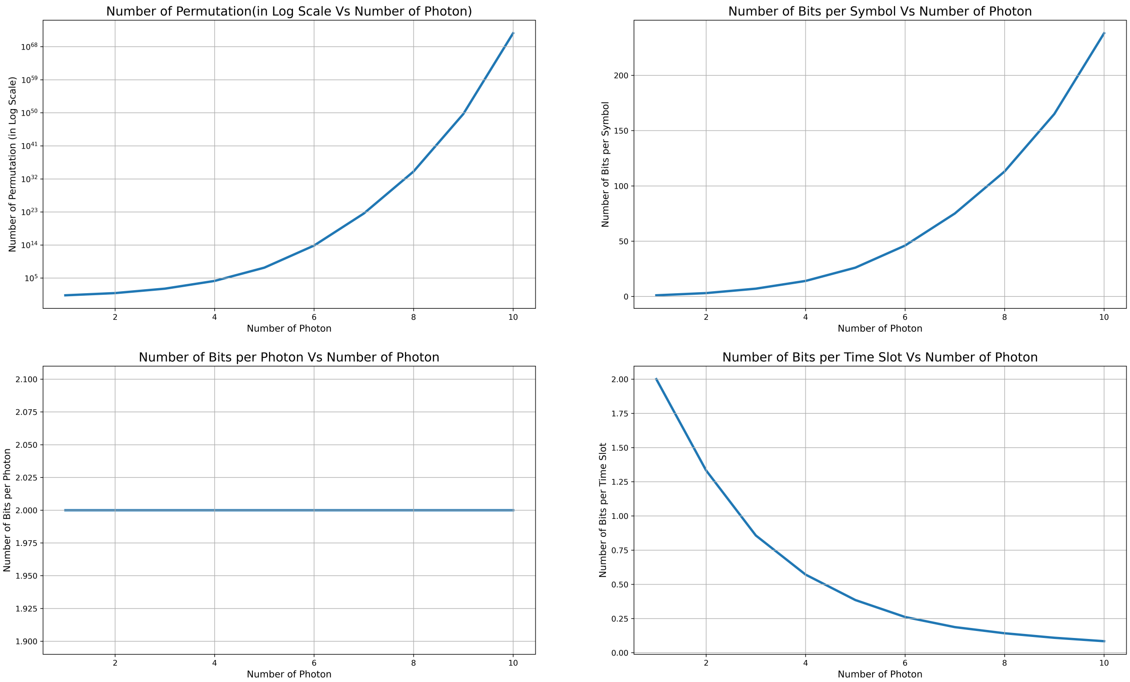

Figure 9.2: Number of Photons Vs Number of Permutation, Number of Photons Vs Number of Bits per Symbol, Number of Photons Vs Number of Bits per Photon, Number of Photons Vs Number of Bits per Time Slot

10

In General

In general, it takes 4 photons in 14 time slots
It means that it has:
1. 1,001 ways to order them
2. 10 bits/symbol
3. 2.5 bits/photon
4. 0.71 bits / time slot

The number of combination is:

$$nCr = \binom{n}{k} = \binom{14}{4} = \frac{14!}{4!(14-4)!} = 1,001$$

	Number of Photon	Number of Permutation	Number of Bits per Symbol	Number of Bits per Photon	Number of Bits per Time Slots
0	1.0	14.0	3.807355	3.807355	0.272
1	2.0	91.0	6.507795	3.253897	0.465
2	3.0	364.0	8.507795	2.835932	0.608
3	4.0	1001.0	9.967226	2.491807	0.712
4	5.0	2002.0	10.967226	2.193445	0.783
5	6.0	3003.0	11.552189	1.925365	0.825
6	7.0	3432.0	11.744834	1.677833	0.839
7	8.0	3003.0	11.552189	1.444024	0.825
8	9.0	2002.0	10.967226	1.218581	0.783
9	10.0	1001.0	9.967226	0.996723	0.712

Figure 10.1: Table

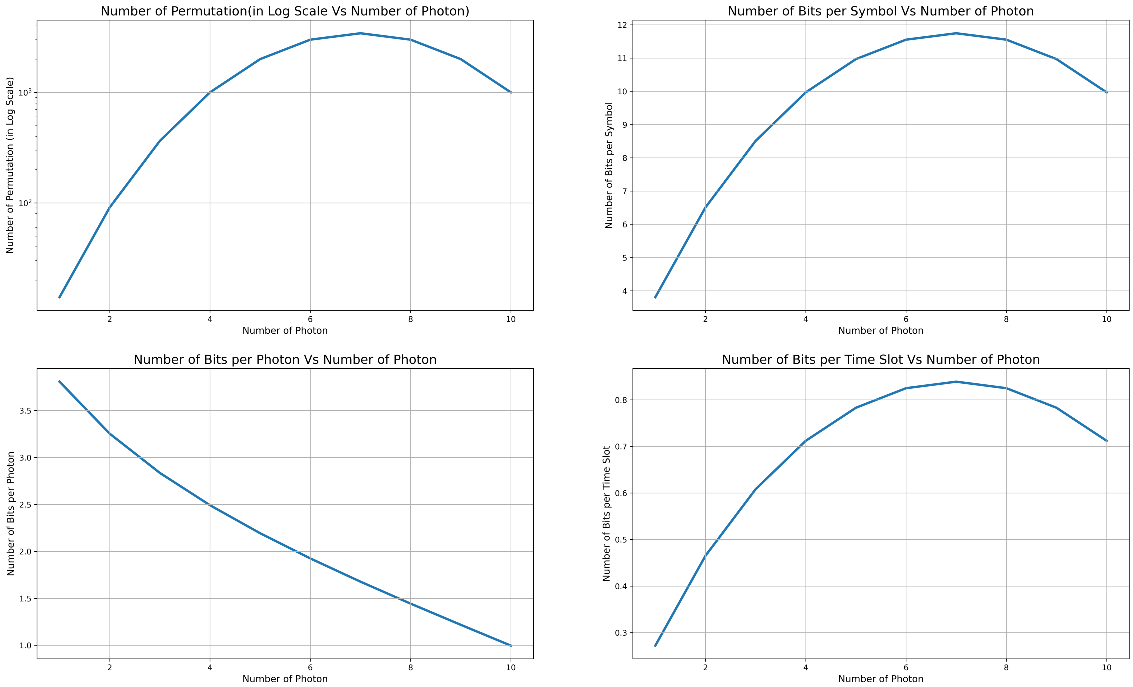

Figure 10.2: Number of Photons Vs Number of Permutation, Number of Photons Vs Number of Bits per Symbol, Number of Photons Vs Number of Bits per Photon, Number of Photons Vs Number of Bits per Time Slot

$$nPr = \binom{n}{k} \times k! = \binom{14}{4} \times 4! = \frac{14!}{(14-4)!} = 24,024$$

figure/General/General_p_Table.png

Figure 10.3: Table

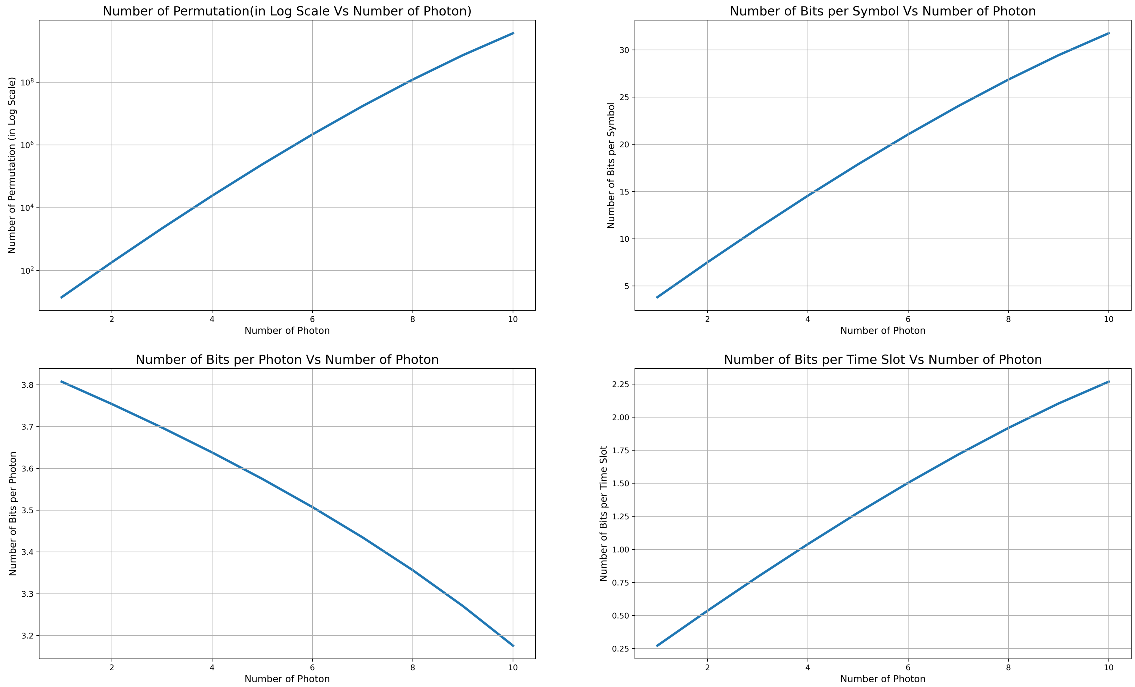

Figure 10.4: Number of Photons Vs Number of Permutation, Number of Photons Vs Number of Bits per Symbol, Number of Photons Vs Number of Bits per Photon, Number of Photons Vs Number of Bits per Time Slot

11

Galois Fields

If p is a prime number, it is possible to define a field with p^m elements for any m. The fields, denoted $GF(p^m)$, are comprised of the polynomials of degree m-1 over the field Z_p. These polynomials are expressed as $a_{m-1}x^{m-1} + \cdots + a_1x^1 + a_0x^0$ where the coefficients a_i take on values in the set $\{0, 1, \ldots, p-1\}$.

$$A_{4\times 3} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix}$$

12

Reed-Solomon Codes

The Reed-Solomon (RS) codes are the non-binary codes, they are important for the use in communication systems where errors appear in bursts rather than independent random errors.

RS codes were discovered by Reed and Solomon in 1960. The encoding process assumes a code of RS(N,K) which results in N code words of length N symbols each storing K symbols of data, being generated, that are then sent over an erasure channel. The non-binary BCH block codes have $2^m(\{0, 1, 2, \ldots, 2^m - 1\})$ symbols with block length $n = 2^m - 1$, which can be extended to $n = 2^m$ or $m = 2^m + 1$. RS codes can correct up to e_0 errors within a block of n symbols by using $n - k = n - 2e_0 = 2^m - 1 - 2e_0$ parity symbols. Or it can locate and correct up to $\frac{t}{2}$ erroneous symbols at unknown locations. As an erasure code, it can correct up to t erasures at location that are known and provided to the algorithm, or it can detect and correct combinations of errors and erasures.

RS code can achieve the maximum number of error correction by finding the largest possible $d_{min} = 2e_0 + 1$

Any combination of K code words received at the other end is enough to reconstruct all of the N code words. The code rate is generally set to $\frac{1}{2}$ unless the channel's erasure likelihood can be adequately modelled and is seen to be less. In conclusion, N is usually 2K , meaning that at least half of all the code words sent must be received in order to reconstruct all of the code words sent.

We construct the encoding matrix E with dimension $k \times n$ where k is the number of check packets and n is the number of data packets.

$$E = \begin{bmatrix} 1 & 1 & 6 \\ 4 & 3 & 2 \\ 5 & 2 & 2 \\ 5 & 3 & 4 \\ 4 & 2 & 4 \end{bmatrix}$$

When the $n \times 1$ vector of data words is multiplied by E, an $k \times 1$ vector of check values is produced.

$$\begin{bmatrix} 1 & 1 & 6 \\ 4 & 3 & 2 \\ 5 & 2 & 2 \\ 5 & 3 & 4 \\ 4 & 2 & 4 \end{bmatrix} \times \begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix} = \begin{bmatrix} 3 \\ 5 \\ 4 \\ 3 \\ 2 \end{bmatrix}$$

The values contained in the 8 'packets' that are sent: (0,4), (1,5), (2,6),(3,3),(4,5),(5,4),(6,3),(7,2). We observe that each of the 8 packets has an identifier that allows the recipient to determine exactly which packets of a Forward Error Correction (FEC) group have been received.

$$V = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 2 & 4 \\ 1 & 3 & 5 \\ 1 & 4 & 6 \\ 1 & 5 & 7 \\ 1 & 6 & 2 \\ 1 & 7 & 3 \end{bmatrix}$$

We have a transformed matrix D from Vandermonde matrix.

$$D = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 6 \\ 4 & 3 & 2 \\ 5 & 2 & 2 \\ 5 & 3 & 4 \\ 4 & 2 & 4 \end{bmatrix}$$

For a collection of n packets including both data and check packets have been received, we extract n rows of the D matrix corresponding the n received packets. We call this $n \times n$ matrix $D^{'}$.

$$D^{'} = \begin{bmatrix} 1 & 1 & 6 \\ 4 & 3 & 2 \\ 5 & 2 & 2 \end{bmatrix}$$

Inverting $D^{'}$ yields $D^{'-1}$.

$$D^{'-1} = \begin{bmatrix} 5 & 5 & 2 \\ 5 & 7 & 3 \\ 3 & 3 & 3 \end{bmatrix}$$

The receiver knows the algorithms by which the check packets were constructed as follow:

$$\begin{bmatrix} 1 & 1 & 6 \\ 4 & 3 & 2 \\ 5 & 2 & 2 \end{bmatrix} \times \begin{bmatrix} d_0 \\ d_1 \\ d_2 \end{bmatrix} = \begin{bmatrix} 3 \\ 5 \\ 4 \end{bmatrix}$$

Multiplying the received check value by D'^{-1} recovers the original values.

$$\begin{bmatrix} 5 & 5 & 2 \\ 5 & 7 & 3 \\ 3 & 3 & 3 \end{bmatrix} \times \begin{bmatrix} 3 \\ 5 \\ 4 \end{bmatrix} = \begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix}$$

Bibliography

[1] Gustaver, M. (2020) A Chalmers University of Technology Master's thesis template for LaTeX. Unpublished.

[2] Ziemer, R., Tranter, W. H. (2006). Principles of communications: system modulation and noise. John Wiley & Sons.

[3] Y. T. Koh and F. M. Davidson, "Interleaved concatenated coding for the turbulent atmospheric direct detection optical communication channel", IEEE Trans. Commun., vol.37, no.6, pp.648-651, 1989.

[4] F. M. Davidson and Y. T. Koh, "Interleaved convolutional coding for the turbulent atmospheric optical communication channel", IEEE Trans. Commun., vol. 36, no. 9, pp.993-1003, 1988.

[5] J. Li and M. Uysal, Optical wireless communications: system model, capacity and coding", IEEE 58th Vehicular Technology Conference (VTC) 2003-Fall, vol. 1, pp. 168-172, 2003.

[6] H. Park and J. R Barry, "Trellis-coded multiple-pulse-position modulation for wireless infrared communications", IEEE Trans. Commun., vol. 52, no. 4, pp. 643-651, 2004.