Analysis and simulation of a digital transmission system

Alessandro Trigolo

2023/2024

Index

Todo list

Introduction

This document analyses and simulates the behavior of a digital transmission system. The document is split into two main parts:

i **Analysis.** `Raw results` section

ii **Simulation.**

You can find the full project at `imAlessas/transmission-simulation.git`.

General schematic

The full schematic - containing every step - of a transmission system is presented in figure 1. Before exploring the mathematical background hidden between the steps, it is crucial to understand what every phase of the system means.

⬦ *Source.* The source device is whichever device is sending a signal; it could be a television, a computer, a smartphone, or anything else.

⬦ *Formatting Device.* The formatting device's task is to translate the information from analogic to digital which translates into sampling the continuous analogic signal and creating a discrete digital signal that can be transmitted through digital devices.

⬦ *Source Coding.* The source coding goal is lossless data compression. Sure enough, through the Shannon-Fano source coding, the symbols transmitted are encoded to reduce the average codeword length.

⬦ *Channel Coding.*

⬦ *Interleaving.* The interleaver is needed to transform package errors into independent errors. This is achieved by changing the ordering of the symbols that will be transmitted.

⬦ *Scrambling.* The scrambling procedure helps with the synchronization between the two devices and improves the security of the transmission. This is achieved by adding a *pseudo-random sequence* to the symbols before the transmission.

⬦ *Modulation.* The modulation process' goal is to match the spectrum of the transmitted signal with the transmission channel bandwidth making the signal more noise-immune and increasing the data-transfer rate; these operations are performed by the modulator. There are different types of modulation, the one utilized in this project is the *Binary Phase Shift Keying*, which is one of the most effective modulations against noise.

⬦ *Noise.* The noise is a crucial obstacle to overcome to have a successful transmission; the noise is the main reason for a wrongly transmitted symbol. There are different types of noise, some of them are generated by other transmissions, others are due to the physical medium and others are caused by the intermediate devices between the transmission. Nevertheless, in every transmission, there will be the *Gaussiam White Noise* which is a thermal noise caused by the Big Bang.
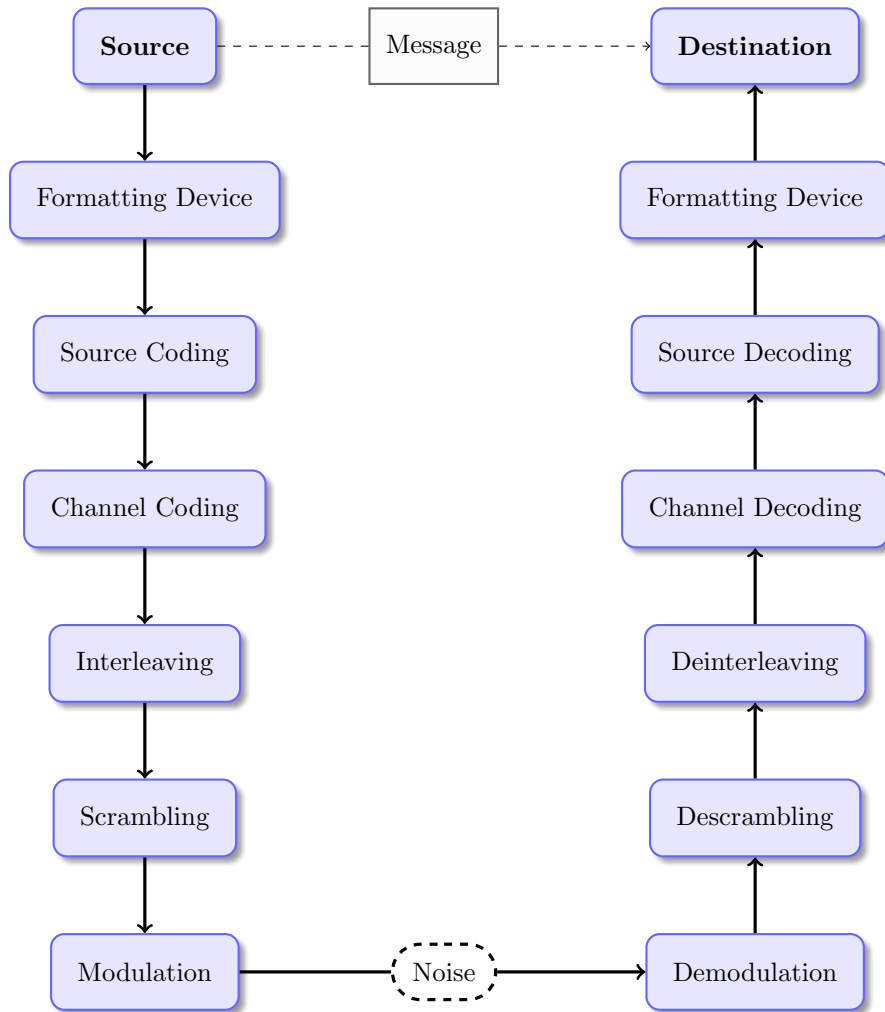
Figure 1: The diagram of the digital information transmission system.

⋄ *Demodulation.* In this phase the demodulator device, after receiving the disturbed signal, will try to detect the signal to regenerate the original one. Sometimes the noise energy will be stronger than the signal energy generating errors that will be corrected in the next steps.

⋄ *Descrambling.* The descrambling procedure is the opposite of the scrambling. The added *pseudo-random sequence*, after the reception is subtracted by the descrambler.

⋄ *Deinterleaving.* The deinterleaver reorders the transmitted symbols in the opposite way that the interleaver did. In such a way the *burst* errors that occurred during the transmission will become single errors that can be easily recovered.

⋄ *Channel Decoding.*

◇ *Source Decoding.* The source decoding procedure decompresses the received data into the original symbols. This is achieved by one of the source coding properties: symbols are easily detected because there are no shorter codes at the beginning of longer codes.

◇ *Formatting Device.* During the transmission this device converts the signal from analogic to digital, during the reception of the signal the formatting device translates the discrete digital signal into a continuous analogic signal.

◇ *Destination.* The destination device is whichever device will receive the signal. Likewise the source one, the destination device could be a satellite, a smartphone, a server, or anything else.

Initial parameters

The parameters used in this project have been assigned in a datasheet and are reported in the following list:

· *Symbol duration*: 60 ns, also called τ;
· *SNR*: 8.1 dB;
· *Source code*: Shannon-Fano coding;
· *Error correction code*: cyclic coding with codeword length $m = 31$ and generator polynomial $z^5 \oplus z^2 \oplus 1$;
· *Carrier frequency*: 2.5 GHz;
· *Modulation*: Binary Amplitude Shift Keying (BPSK) with the phase shift of π.

In addition, the source data (alphabet) and the symbols' respective probabilities are summarized in the following table.

Source 7	
a_1	0.11
a_2	0.07
a_3	0.09
a_4	0.01
a_5	0.06
a_6	0.06
a_7	0.13
a_8	0.14
a_9	0.13
a_{10}	0.05
a_{11}	0.11
a_{12}	0.04

Afterwards the parameters have been transcripted in the MATLAB program, as shown in the following snippet.

```
% source number 7
ALPHABET = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12];
PROBABILITY_VECTOR = [11, 7, 9, 1, 6, 6, 13, 14, 13, 5, 11, 4]/100;

```

```matlab
TAU = 60e-9; % symbol duration time, [s]
SNR = 8.1; % Signal-to-Noise-Ration, [dB]
% Source Code: Shannon-Fano
% Error correction code: Cyclic
CODEWORD_LENGTH = 31; % m
%
F_0 = 2.5e+9; % carrier frequency [Hz]
% Modulation: BPSK
PHASE_SHIFT = pi; % [rad]
U = 1; % amplitude BPSK signal [V]

transmitted_symbol_number = 20;
```

Part I
Analysis

1 Source data

The source data analysis provides a general overview of how the data are generated and how this will impact the encoding scheme. Specifically, the source data analysis is achieved by calculating two important values: the source entropy and the redundancy coefficient.

Source entropy

The source entropy H and the maximum source entropy H_{max}. The entropy of a sequence of symbols is a number that summarizes the randomness of the selection of the symbols in the source sequence. The more uncertain the symbols are, the higher the entropy is and the higher the information the symbols carry. The ideal entropy is when the source symbols are `1 0 1 0 1 0 ...` while the worst entropy is when all the symbols are `1` or `0`. Given a sequence S of N symbols, where each of them has its probability P_i to occur, the entropy of the sequence is:

$$H(S) = -\sum_{i=1}^{N} P_i \log_2 P_i$$

The entropy calculation can be simply achieved with the following MATLAB code. The only thing to note is that P is the probability vector that assigns to every symbol of the alphabet its probability.

```
% sum(V .* log2(V))
H = - dot(PROBABILITY_VECTOR, log2(PROBABILITY_VECTOR));
```

Secondly, in order to calculate the maximum entropy H_{max}, two conditions have to be met: all of the symbols have the same probability $P_i = \frac{1}{N}$ and, of course, they do not correlate one another. Consequently:

$$H_{max}(S) = -\sum_{i=1}^{N} \frac{1}{N} \log_2 \frac{1}{N} = \frac{1}{N} \sum_{i=1}^{N} \log_2 N = \log_2 N$$

Also in this case the MATLAB script to calculate the maximum entropy is trivial.

```
% Number of symbols in the alphabet
N   = length(PROBABILITY_VECTOR);

% Maximum source entropy
H_max = log2(N);
```

By running the scripts, the value obtained are $H = 3.3995$ while $H_{max} = 3.5850$. Reasonably $H < H_{max}$ because the given probabilities in the datasheet weren't equal to each other.

Redundancy coefficient

The redundancy coefficient ρ summarizes in a number how much additional information is present inside the sequence. Essentially, the lower the redundancy coefficient is, the better, because it means that the source entropy is very high. Mathematically, the coefficient ρ can be obtained as follows:

$$\rho = 1 - \frac{H}{H_{max}}(S)$$

which translates into the following code snippet:

```matlab
% Calculate the redundancy coefficient 'rho'
source_redoundancy = 1 - H/H_max;
```

Expectedly, the redundancy coefficient is not zero because $H < H_{max}$: by running the script, $\rho = 0.0517$.

2 Source encoding

The source coding analysis provides the necessary tools to evaluate the source coding algorithms for efficient data representation and compression. In this case, the analysis calculates and uses different values to provide a better understanding of the efficiency of the Shannon-Fano source coding. Particularly the values that will be analyzed are the average codeword length $\overline{m}$, the probability of 1 and 0 (P_1 and P_0), the binary entropy H_{bin}, the source data generation rate R and the compression ratio K.

Shannon-Fano algorithm

Before calculating the values it is important to encode the symbols of the alphabet through the Shannon-Fano algorithm. A brief recursive description of it is reported below.

1. Sort the symbol of the alphabet by descending probability;

2. Divide the sets of symbols into two continuous subsets with the same probability (or the lowest difference between the two);

3. Assign to one subset the symbol 1 and the other 0;

4. Repeat until every subset consists of one symbol;

5. Read the codeword from left to right.

By applying the Shannon-Fano algorithm to the given source, the result should be the following.

S	P	Shannon-Fano					Code	m	m_0	m_1
a_8	0.14	1	1	1			111	3	0	3
a_7	0.13			0			110	3	1	2
a_9	0.13		1	1			101	3	1	2
a_1	0.11			0			100	3	2	1
a_{11}	0.11			1			011	3	1	2
a_3	0.09	0	1	0	1		0101	4	2	2
a_2	0.07				0		0100	4	3	1
a_5	0.06			1	1		0011	4	2	2
a_6	0.06				0		0010	4	3	1
a_{10}	0.05		0		1		0001	4	3	1
a_{12}	0.04			0	0	1	00001	5	4	1
a_4	0.01					0	00000	5	5	0

After computing the Shannon-Fano algorithm to the given source, the results should be inserted into the MATLAB program, as follows.

```matlab
% Probability vector sorted from highest to lowest
P = sort(PROBABILITY_VECTOR, 'descend');


% Values obtained with Shannon-Fano code algorithm

% Symbols codeword length
m = [3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 5, 5];

% Number of 0s inside the symbols codeword
m_0 = [0, 1, 1, 2, 1, 2, 3, 2, 3, 3, 4, 5];

% Number of 1s inside the symbols codeword
m_1 = [3, 2, 2, 1, 2, 2, 1, 2, 1, 1, 1, 0];
```

Binary entropy

At this point, there is all the needed information to calculate the required data for the analysis. First of all, to calculate the average codeword length $\overline{m}$ of N symbols, the following formula should be computed:

$$\overline{m} = \sum_{i=1}^{N} m_i \cdot P_i$$

Additionally, to calculate P_0 and P_1, it is necessary to calculate also the average number of 0 and 1 symbols. The formula is the same as for the average codeword length:

$$\overline{m_0} = \sum_{i=1}^{N} m_{0_i} \cdot P_i \qquad \overline{m_1} = \sum_{i=1}^{N} m_{1_i} \cdot P_i$$

After inserting these formulas in the MATLAB script, the values are $\overline{m} = 3.4300$, $\overline{m_0} = 1.6400$ and $\overline{m_1} = 1.7900$.

```matlab
% Average codeword length
m_average = dot(P, m);

% Average number of 0 symbols
m_0_average = dot(P, m_0);

% Average number of 1 symbols
m_1_average = dot(P, m_1);
```

Moreover, by dividing the number of 0 or 1 symbols by the average length of the codeword the two probabilities, P_0 and P_1, can be computed:

$$P_0 = \frac{\overline{m_0}}{\overline{m}} \qquad P_1 = \frac{\overline{m_1}}{\overline{m}}$$

The two probabilities values are $P_0 = 0.4781$ and $P_1 = 0.5219$. Ideally, the probabilities should be $P_0 = P_1 = 0.5$; nonetheless, the two values are still very close to each other. Finally, with the two probability values the binary entropy H_{bin} can be obtained using the following calculation.

$$H_{bin}(S) = -P_0 \log_2 P_0 - P_1 \log_2 P_1$$

By running the following MATLAB script, the value of the binary entropy is $H_{bin} = 0.9986$ which is very close to 1. The higher the entropy is, the more uncertainty is associated with every symbol: this means that encoding the initial data with the Shannon-Fano algorithm provides a great value, information-wise.

```matlab
% Probability of 0 symbol
P_0 = m_0_average / m_average;

% Probability of 1 symbol
P_1 = m_1_average / m_average;

% Binary source entropy after coding
H_bin = -P_0 * log2(P_0) - P_1 * log2(P_1);
```

Data rate and compression ratio

After encoding the source data with the Shannon-Fano algorithm, it is important to evaluate the source data generation rate R, which can be calculated as follows:

$$R = \frac{H(S)}{\overline{m}\tau}, \text{ where } \tau \text{ is the symbol duration}$$

The data compression ratio K is important as well: it helps evaluate how much the initial data has been compressed after the source coding. The following formula will help to obtain this value.

$$K = \frac{\overline{m}}{H(S)}$$

After running the MATLAB script displayed below, the data rate $R = 16.519 Mbit/s$ which should be lower than the channel capacity C_{chan} with noise. Moreover, the compression ratio $K = 1.0090$ which is very close to 1, means that the overall compression is low: this is still not a bad result because the overall entropy is increased significantly after the source coding.

```matlab
% Calculate Data Rate
R = H * (m_average * TAU) ^ (-1);

% Calculate Compression Ratio
K = m_average / H;
```

3   Shannon's theorem condition

Raw results

Make a little introduction in the "Raw result" section

Symbol	Description	Value
	Task 2	
$H(S)$	Source entropy	3.3995
$H(S)_{max}$	Maximum source entropy	3.5850
ρ	Source redundancy	0.0517
	Task 3	
$\overline{m}$	Average codeword length	3.4300
P_0	Probability of "0"	0.4781
P_1	Probability of "1"	0.5219
$H_{bin}(S)$	Binary entropy	0.9986
R	Source data rate	16.519 Mbps
K	Compression ratio	1.0090
	Task 4	
C_{bin}	Noiseless channel capacity	16.667 Mbps
P_{err}	Error probability (BER)	$1.6315 \cdot 10^{-4}$

Conitnue on next page . . .

Symbol	Description	Value
C_{chan}	Channel capacity with noise	16.629 Mbps
Task 5		
Task 6		
Task 7		
Task 8		