# Project # 3. Multimedia Coding

Alexandre Rodrigues (2039952)

January 12, 2022

## 1  Introduction

This reports is dedicated to explain the usage of the LBG-split algorithm and how to implement it. The Linde-Buzo-Gray algorithm is a lossy coding technique, it uses vector quantization, meaning that a block of input samples (size L) is processed together. . . .

## 2  Technical Approach

Each vector of size L samples can be defined as

$$x = [x_1, x_2, \ldots, x_L], x \in R^L \tag{1}$$

where $x_i, i = 1, 2, \ldots, L$ are input samples. Using $y_i$ as a codevector,

$$B = \{y_1, y_2, \ldots, y_L\} \tag{2}$$

is the set of reconstruction levels, i.e. the codebook, of size K. The decision cells will be

$$I_i \in R^L, i = 1, 2, \ldots, K, such that I_i intersect I_j = 0 \aleph i \neq j and union k i = 1 I_i = R^L \tag{3}$$

### 2.1  LBG

Since we do not know the probablity distributuion function of the input data $f_x(x)$ we can use a training set

$$T = x_1, \ldots, x_N \tag{4}$$

, where $N$ should be considerably larger than the codebook size $K$,$N \leq 500K$ for this work.

The algorithm can then be defined as:

1. initial codebook

2. optimal partition

3. new codebook

4. distortion

5. terminate?

## 2.2  Split approach

This approach is based on starting a codebook as

$$\{(1 - \epsilon)y_{avg}, (1 + \epsilon)y_{avg}\}, \tag{5}$$

where $y_{avg}$ is the average of the vectors in the training set. The LBG algorithm is then applied to this codebook. The returning optimized codebook is split in the same way, i.e.

$$\{(1 - \epsilon)y_i, (1 + \epsilon)y_i, \ldots, (1 - \epsilon)y_N, (1 + \epsilon)y_N\}, \tag{6}$$

This implies that the codebook size will double in each iteration until we get the desired size K, $N = 2, 4, 8, \ldots, K$.

# 3  Results

There were several tests made to fully understand the performance of this method. There 3 different trainng sets used:

- All audio files from MC dataset

- Only the Say Nada song

- All Audio Files and All Popular Music

Each training set was limited in size. For each file I extracted the middle part to result in the following relative size (N/K).

| Training set | N/K for L = 2 | N/K for L = 4 |
|---|---|---|
| All Audio | 1250 | 625 |
| Say Nada | 1000 | 500 |
| All Music and Audio | 4844 | 2422 |

Table 1: Relative Sizes of each Training Set

These sizes allowed fast enough codebook computation. Having training sets with and without including the encoding objects will benefit our comparison and possible conclusions.

## 3.1  Training Performance

The tests were made using $\epsilon = 0.01$.

| Training set | 2,2 | 2,4 | 4,1 | 4,2 |
|---|---|---|---|---|
| All Audio | $1.51 \times 10^5$ | $3.53 \times 10^5$ | $4.48 \times 10^5$ | $3.15 \times 10^6$ |
| Music: Say Nada | $4.36 \times 10^6$ | $?? \times 10^6$ | $3.31 \times 10^7$ | $?? \times 10^7$ |
| All Music and Audio | $2.47 \times 10^6$ | $?? \times 10^6$ | $9.70 \times 10^6$ | $?? \times 10^7$ |

Table 2: Distortion for each training set and each values of L and R

| Training set | 2,2 | 2,4 | 4,1 | 4,2 |
|---|---|---|---|---|
| All Audio | $1.29s$ | $179.09s$ | $0.45s$ | $86.025s$ |
| Music: Say Nada | $0.75s$ | $s$ | $0.47s$ | $s$ |
| All Music and All Audio | $2.57s$ | $??$ | $1.21s$ | $??$ |

Table 3: Time for each training set and each values of L and R

We can see that (4,1) is clearly the fastest training. The training time is mostly dependent on the value K. The distortion is noticeably larger for L = 4.

## 3.2 Encoding Performance

In summary I got the following results:

| Encoded | 2,2 | 2,4 | 4,1 | 4,2 |
|---|---|---|---|---|
| 70mono | $1.20s$ | $16.79s$ | $0.60s$ | $8.705s$ |
| Average Music | $13.40s$ | $193.61s$ | $6.66s$ | $104.92s$ |
| Worst Case | $18.40s$ | $232.67s$ | $8.22s$ | $118.26s$ |
| Best Case | $10.89s$ | $149.41s$ | $5.52s$ | $91.04s$ |

Table 4: Time for each training set and each values of L and R

There is no noticeable difference when using All Audio or Say Nada as the training set.

| Encoded | 2,2 | 2,4 | 4,1 | 4,2 |
|---|---|---|---|---|
| 70mono | $5 \times 10^5$ | $2 \times 10^4$ | $4.75 \times 10^5$ | $8 \times 10^4$ |
| Average Music | $3.72 \times 10^6$ | $5.77 \times 10^5$ | $4.66 \times 10^6$ | $8.45 \times 10^5$ |
| Worst Case | $1.15 \times 10^7$ | $8.20 \times 10^5$ | $1.36 \times 10^7$ | $2.74 \times 10^6$ |
| Best Case | $6.2 \times 10^5$ | $3.10 \times 10^4$ | $9.42 \times 10^5$ | $1.07 \times 10^5$ |

Table 5: Distortion for each training set and each values of L and R

Encoding music files using All Audio as training se gives us in average 4 times the distortion when using Say Nada as the training set. For (2,4) we get actually 11 times the distortion.

## 3.3 Training set including the encoding object...

# 4 Conclusions

1 - Using music as a training set is clearly superior when we will use the codebook to encode music files. Vice-versa is also valid, although less significantly. Encoding 70mono using All audio as training set produces in average half the distortion.

2 -

3 -

4 -