

# Bit-Plane Encoding for the Differential Image in DPCM

# 1. Introduction

- The differential image in DPCM typically has a largely reduced variance and significantly reduced correlation compared to the original image.
- A majority of the pixel values in the differentially encoded image is close to zero, as shown in Fig. 7.3.

- Binary representations of these pixel values are expected to have more ‘0’ bits than ‘1’ bits in more significant bits (MSBs). ‘1’ bits are more concentrated at the less significant bits.
- Adaptive arithmetic coder uses an adaptive probability model that matches the local probability distribution. It is found to outperform the fixed model version in terms of compression efficiency.

## 2. Arithmetic Coding

### 2.1. Arithmetic Coding with Fixed Model

Arithmetic coding is a non-block code, where each symbol  $s_i$  in the source string is coded with a codeword length approximately equals  $-\log_2 p(s_i)$ .

Given a source symbol set  $S = \{s_1, \dots, s_n\}$  with probabilities  $P = \{p_1, \dots, p_n\}$  correspondingly, the arithmetic coding algorithm can be described with the help of the following affine transformation

$$w_i : [0, 1] \rightarrow [0, 1]$$

where

$$w_i(x) = p_i x + t_i, \quad i = 1, \dots, n, \quad (1)$$

with

$$t_i = \sum_{k=1}^{i-1} p_k, \quad \text{and } t_1 = 0.$$

The arithmetic coding procedure to encode a given source stream  $s_{i_1}s_{i_2}\cdots s_{i_m}$  of  $m$  symbols is:

**Step 1**  $x = 0, k = 1.$

**Step 2** Apply affine transform to  $x$ :

$$y = w_{i_k}(x) = p_{i_k}x + t_{i_k}.$$

**Step 3**  $k = k + 1.$

If  $k \leq m,$

then

$$x = y,$$

goto step 2.

$y$  is therefore the number representing the source stream.

## 2.2. Adaptive Arithmetic Coding

An *adaptive model* represents the changing symbol frequencies (probabilities) seen so far in a source string. Such a model matches the local statistics in the source string. In practice, it outperforms the fixed model version in terms of compression efficiency.

The adaptive algorithm to encode the source string  $s_{i_1}s_{i_2}\cdots s_{i_m}$  of  $m$  is as follows:

**Step 1** Initialize the source model.

Given the source symbol set  $S = \{s_1, \dots, s_n\}$ , initialize the frequency of each symbol to a constant,  $f_i = f_0$ ,  $i = 1, \dots, n$ .

**Step 2**  $k = 1$ ,  $x = 0$ .

**Step 3** Compute the probability for each source symbol:

$$p_i = \frac{f_i}{\sum_{j=1}^n f_j}.$$

**Step 4** Apply affine transform to  $x$ :

$$y = w_{i_k}(x).$$

**Step 5** Update source model:  $f_{i_k} = f_{i_k} + 1$ .

If  $\sum_{j=1}^n f_j == MAXFREQUENCY$

$f_j = f_j/2, j = 1, \dots, n$ .

**Step 6**  $k = k + 1$ .

If  $k \leq m$ ,

then

$x = y$ ,

goto step 3.

## **REFERENCE:**

I. H. Witten, R. M. Neal, and J. Cleary, "Arithmetic Coding for Data Compression," *Commun. ACM*, Vol. 30, No. 6, 1987.

### 3. Bit-Plane Ordered Representation of the Differential Image in DPCM

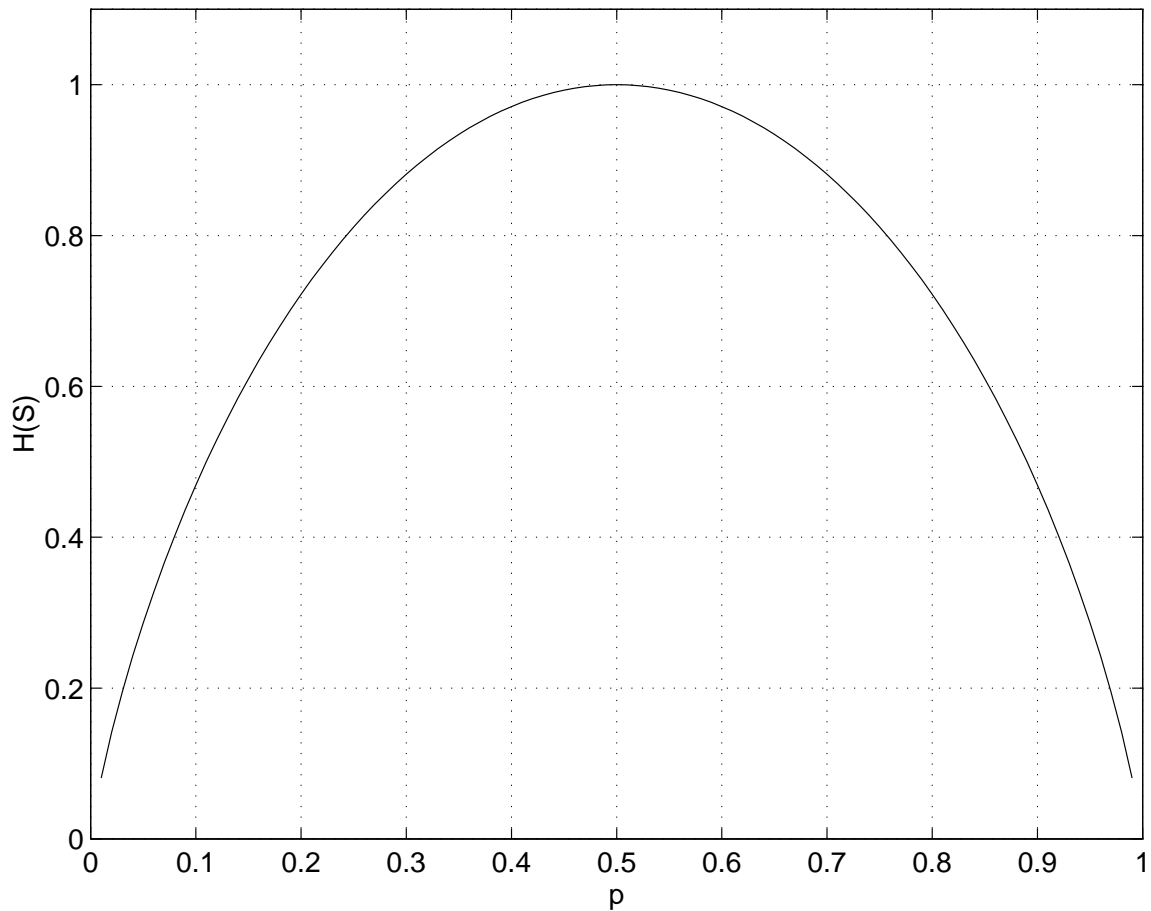
- The *entropy* of a source  $S = \{s_1, s_2, \dots, s_n\}$  is given by:

$$H(S) = - \sum_{i=1}^n p(s_i) \log_2 p(s_i) \quad (2)$$

where  $p(s_i)$  is the probability that symbol  $s_i$  occurs in the source stream.

- Entropy  $H(S)$  is the average amount of information per source symbol provided by the source, measured in bit/symbol. It can also be interpreted as the average cost to encode a symbol in the source stream.

- It is easy to prove that the maximum entropy occurs when all symbols have the same probability. The entropy is lower when the symbols' probabilities are different.



Change in Entropy with the probability  
distribution in two-symbol case

$$H(S) = -p \log_2 p - (1 - p) \log_2(1 - p)$$

- The adaptive arithmetic coder uses an adaptive probability model, in which the probability of each symbol is determined by the probability distribution of its predecessors.
- If the symbols in the source stream can be ordered in such a way that the local probability distribution is heavily in favor of a certain symbol, the total cost of encoding the source stream can be reduced.
- Binary representations of the pixel values in the differential image of DPCM are expected to have more '0' bits than '1' bits in more significant bits (MSBs). '1' bits are more concentrated at the less significant bits.

- In the differential image of DPCM, the pixel value is in the range  $[-255, +255]$ . Each value  $x^i$  can be represented in a 9-bit binary string,  $s_0^i, b_1^i, b_2^i, \dots, b_8^i$ , including the sign bit. The bit-plane ordering of the values are:

$$s_0^1, s_0^2, \dots, s_0^n,$$

$$b_1^1, b_1^2, \dots, b_1^n,$$

$$\vdots$$

$$b_8^1, b_8^2, \dots, b_8^n.$$

The ordered binary string is then encoded by the adaptive arithmetic coder.

## Example of bit-plane ordering:

Assume a sequence of pixel values:

0, 16, -3, 24, 50, -20, 6, 100, -7.

The binary representations of each value are:

0 : 0 0 0 0 0 0 0 0 0 0

16 : 0 0 0 0 1 0 0 0 0 0

-3 : 1 0 0 0 0 0 0 0 1 1

24 : 0 0 0 0 1 1 0 0 0 0

50 : 0 0 0 1 1 0 0 1 0 0

-20 : 1 0 0 0 1 0 1 0 0 0

6 : 0 0 0 0 0 0 1 1 0 0

100 : 0 0 1 1 0 0 1 0 0 0

-7 : 1 0 0 0 0 0 1 1 1 0

It is easy to see that bit-plane ordering will result in much higher probability for '0' bits than for '1' bits in MSBs (to about bit 4 in this example, exclude the sign bits).