# APE 无损压缩算法

谁知脂砚是湘云

Chineseguuys@github

NanJing China –2021 年 10 月 10 日

## 数字音频

声音是一种波。数字音频是波的一种数字表示形式。通过每秒钟对模拟的声音信号的幅度值进行多次采样来获得音频的数字表示形式。也就是说，每一秒采样多次波形的高度。今天的音频 CD 可以在每一秒种采样 44100 个样本。由于 CD 存储的是立体声，所以它每一秒中需要分别采样 44100 个左右声道的样本。在比较流行的采样精度中，这些值是 16bits 的整数值。对于一个 WAV 文件，记录了连续的 L,R,L,R 的采样数据，由于每一个样本需要占用 32 bits (左声道 16 bits，右声道 16 bits)。并且每一秒钟有 44100 个样本。所以一秒钟的音频需要占用 1,411,200 位或 176,400 字节。

## 无损编码

无损压缩可以分为下面的几个步骤。详述如下。

### 转换 $x, y$

无损压缩的第一步是将左右声道的值 $L$ 和 $R$ 转化为更加有效的 $X$ 和 $Y$ 值。$L$ 和 $R$ 通道值之间往往存在相关性。可以通过多种方式来利用这一点。其中一种流行的方式是使用中值/边缘编码方式。中间值 ($X$) 是两个声道的平均值，边缘值 ($Y$) 是两个声道之间的差分值。可以这样实现：

$$X = L - R$$
$$Y = \frac{L + R}{2}$$

### 预测器

接下来，$X$ 和 $Y$ 数据通过预测器以尝试去除任何冗余。基本上，此阶段的目标是使 $X$ 和 $Y$ 数组包含尽可能小的值，同时仍保持可解压 (预测器的算法需要保持可逆)。几乎有无数种方法可以做到这一点。这是一个使用简单线性代数的示例：

- $P_X$ 和 $P_Y$ 是 $X$ 和 $Y$ 的预测值；$X_{-1}$ 是 $X$ 的上一时刻的值，$Y_{-1}$ 是 $Y$ 的上一个时刻的值,$X_{-2}$ 是 $X$ 上两个时刻的值
- $P_X = 2X_{-1} - X_{-2}$

- $P_Y = 2Y_{-1} - Y_{-2}$

举一个简单的例子，如果我们有序列 $X = \{2\ 8\ 24\ \}$；那么对当前时刻的预测值 $P_X = 2 \cdot 24 - 8 = 40$。之后我们的预测值和实际值进行比较，它们之间的差值将会被送入下一个阶段进行编码。

好的预测器都是自适应的，因为它们会根据当前数据的“可预测性”进行调整。例如，我们可以使用一个值从 0 到 1024 的因子 $m$(0 表示没有预测，1024 表示完全预测)。每一次预测之后，$m$ 会根据预测的值是否有用向上或者向下调整。所以在前面的例子当中，剩下的预测器是这样的：

- $X = \{2, 8, 24, ?\}$
- $P_X = 2X_{-1} - X_{-2} = 40$

如果此时 $? = 45$，$m = 512$，那么

$$final_v = ? - \left(P_X \cdot \frac{m}{1024}\right) = 45 - \left(40 \cdot \frac{512}{1024}\right) = 25$$

在此之后，$m$ 的值将会向上进行调整，因为更大的 $m$ 将会更加的有效。

### 数据编码

压缩的目标是通过尽可能的消除数据之间的冗余来使得所有的数据尽可能的小。为什呢数据越小越好呢？因为小的数字可以用少量的比特进行表示。比如说，针对 32 位的无符号整数数组 $10, 14, 15, 16$，它们的二进制表示形式为：$1010, 1110, 1111, 101110$。可以很明显的看出来，如果每一个数字都使用 32 位来进行表示将会非常的消耗空间。理想的做法是分别使用它们最短的二进制表示形式，组合起来就是 101011101111101110。但是这里存在的问题就是我们不知道每一个数字从什么地方开始，到什么地方结束。

为了存储比较小的数字，让它们可以占用更少的位数，同时保证它们是可以解压缩的，可以使用“熵编码”。下面详细介绍了被称之为 Rice Coding 的熵编码系统。Monkey Audio 使用了一种更加先进的熵编码系统。但是 Rice Coding 的基础的知识依然适用。

## 熵编码

Rice 使用比较少的比特数来表示比较小的数，同时保持区分每一个数字的能力。他基本上是这样工作的：

1. 你对一个数字需要多少位来进行编码进行预测，并将预测值记做 $k$
2. 取最右侧的 $k$ 个比特的数字并记住它。
3. 舍弃最右侧的 $k$ 个比特，查看剩余部分的值
4. 对这个数字进行编码，编码包含了步骤 3 中数字个数的 0，一个终止 1，数字最右侧的 $k$ 个比特

**例子**

让我们看一个例子，比如说对于数字序列 $10, 14, 15, 46$ 当中的最后一个数字进行编码：

1. 对数字的位数进行预测；由于前面的三个数字都是使用四个比特进行编码，所以可以预测第四个数字也是使用 4 个比特进行编码，即 $k = 4$
2. 取出数字 46(101110) 的最后 $k$ 个比特 1110
3. 舍弃最右侧的 $k$ 个比特，查看剩余的部分是 (10), 十进制值是 2
4. 我们对数字 46 进行编码，首先是 2 个 0，然后是一个终止 1，最后是 $k$ 个比特的 1110。最终的编码结果是 0011110

在数学上具有更加简洁的表达形式：假设某一个整数 $n$ 是需要编码的数字，而 $k$ 是预计的编码的位数；

1. 符号位 (1 表示正数，0 表示复数)
2. 需要 $\frac{n}{2^k}$ 个 0
3. 一个终止 1
4. $n$ 的最右边的 $k$ 个比特