PVQ Encoding with Non-Uniform Distribution

Jean-Marc Valin, Timothy B. Terriberry

July 6, 2015

1 Introduction

The pyramid vector quantizer (PVQ) is common form of algebraic vector quantization. It is useful in the context of both audio and video compression. The PVQ codebook is defined

$$S\left(N,K\right)=\left\{\mathbf{y}\in\mathbb{Z}^{\mathbb{N}}:\sum_{i=0}^{N-1}|y_i|=K\right\}\ , \tag{1}$$

the set of all integer vectors in N dimensions for which the sum of absolute values equals K. When all codevectors are considered to have equal probability, several methods [2, 3] exist to convert between any codevector and an index J in the range $[0,\, V(N,K)-1]$, where $V(N,K)$ is the number of elements in $S(N,K)$. The index is then easily coded in a bit-stream, possibly with the use of a range coder [4] to allow for fractional bits since $V(N,K)$ is generally not a power of two. The equal-probability case is common for audio. For video, transform coefficients (e.g. DCT) or any prediction residual for a block tend to have widely different distributions. For this reason, using a uniform probability model. This document proposes a way to efficiently encode the result of PVQ quantization with such non-uniform distributions.

2 Non-Uniform Distribution

The non-uniform probability distribution of the codevectors requires building a probability model. For any codebook of reasonable size, explicitly modelling the distribution of J itself is impractical since $V(N,K)$ can easily exceed 32 bits. Instead, we use parametric models for the distribution of $|y_i|$ as a function of i. Sections 2.1 and 2.2 present two possible models for encoding non-uniform PVQ parameters. Both models assume the use of a range/arithmetic coder, ideally one that is capable of encoding non-binary symbols. In most cases, the probability distribution functions (pdf) can be stored in a lookup table in the form of cumulative distribution functions (cdf) that can be used directly by the encoder and decoder.

2.1 Coefficient Magnitude Model

The coefficient magnitude (CM) model is based on the expected absolute value of the coefficient i

$$\sigma_i = E\{|y_i|\} = \sum_{k=0}^{\infty} p_i(k) \ , \tag{2}$$

where $p_i(k)$ is the probability that $|y_i| = k$. We assume that y is the result of quantizing x to the nearest integer, where x follows a Laplace distribution

$$p(x) = r^{-|x|} \ . \tag{3}$$

Assuming the positive quantization thresholds are $\theta + k$, $k \in \mathbb{N}$, we have

$$p(k) = \begin{cases} 1 - r^{\theta} & , \ k = 0 \\ r^{\theta}(1-r)\, r^{k-1} & , \ k \neq 0 \end{cases} \ . \tag{4}$$

The value of r is obtained by modelling σ_i. By assuming $\theta = 1$, we can have a simple relation for r

$$r = \frac{\sigma_i}{1 + \sigma_i} \ . \tag{5}$$

We can still use $\theta \neq 1$ to model $p(k)$ itself, in which case (5) becomes an approximation. Typically, θ is in the range $\left[\frac{1}{2}, 1\right]$. For efficiency reasons, we pre-compute the cdf corresponding to $p(k)$ for different values of r.

If all values y_i are identically distributed, then all expectiations σ_i are equal and simply $\sigma_i = K/N$. In practice, we assume that the values y_i are in decreasing order of expected value and make the approximation

$$\sigma_0 = \alpha K/N \ , \tag{6}$$

where α represents how uneven the distributions are ($\alpha = 1$ corresponds to identical distributions). Knowing α, we can obtain σ_0, r_0 and thus $p_0(k)$, making it possible to encode (and decode using the same process) y_0. Knowing the value of y_0, we can encode y_1 using

$$\begin{aligned} N^{(1)} &= N - 1 \\ K^{(1)} &= K - |y_0| \ . \end{aligned} \tag{7}$$

The process can be applied recursively until $K = 0$ or $N = 1$. The coefficient α is assumed constant across a vector and adapted between vectors based on the observed expectation σ_i as a function of K and N. Similar to a linear regression, we have

$$\alpha = \frac{S_y}{S_E} \ , \tag{8}$$

where for new vector, we update S_y and S_E as

$$S_y \leftarrow (1-\eta)\, S_y + \eta \sum |y_i| \tag{9}$$

$$S_E \leftarrow (1-\eta)\, S_E + \eta \sum \left(K^{(i)}/N^{(i)} \right) \ , \tag{10}$$

where η controls the adaptation rate.

The total number of symbols coded with this approach is equal to the position i_{last} of the last non-zero component of $\mathbf{y}$.

2.2 Run-Length Model

For long sparse vectors, the method in Section 2.1 is inefficient in terms of symbols coded. In those cases the run-length (RL) model models $q(n)$, the probability of y_n being the first non-zero coefficient in $\mathbf{y}$, as a truncated exponential distribution

$$q\left(n\right) = C_1 \begin{cases} r^{-n} & , \; n < N \\ 0 & , \; n \geq N \end{cases} , \tag{11}$$

where $C_1$is a normalization constant. We then model the expected value of $q(n)$ as

$$\sigma_n = E\left[q\left(n\right)\right] = \beta \cdot \frac{N}{K} , \tag{12}$$

where $\beta = 1$ represents the case where non-zero coefficients are distributed evenly in the vector (typically, $\beta < 1$).

The relationship between σ_n and r is given by

$$\sigma_n = \frac{r^N - Nr + N - 1}{r^{N-1}\left(1-r\right)^N} , \tag{13}$$

so computing r from σ_n is not easy. We use the approximation

$$r \approx \frac{\sigma_n}{1+\sigma_n} + \frac{8\sigma_n^2}{\left(N+1\right)\left(N-1\right)^2} . \tag{14}$$

Once the position n of the first non-zero coefficient is coded, one pulse is subtracted from that position and the process is restarted with

$$\begin{aligned} N^{(1)} &= N - n \\ K^{(1)} &= K - 1 . \end{aligned} \tag{15}$$

If multiple pulses are present at a certain position, then we encode a position of zero for each pulse that follows the first pulse.

Because the sign is fixed once a pulse is already present at a certain position, the probability of adding a pulse is divided by two. The distribution then becomes

$$q(n) = C_2 \begin{cases} 1/2 & , \; n = 0 \\ r^{-n} & , \; 0 < n < N \\ 0 & , \; n \geq N \end{cases} . \tag{16}$$

The β parameters is adapted in a similar way as the α parameter in 2.1:

$$\beta = \frac{S_p}{S_N} , \tag{17}$$

where for each new vector, we update S_p and S_N as

$$S_p \leftarrow (1-\eta)\, S_p + \eta \sum n^{(i)} K^{(i)} \tag{18}$$

$$S_N \leftarrow (1-\eta)\, S_N + \eta \sum N^{(i)} \ , \tag{19}$$

where η controls the adaptation rate.

2.3 Model Combinations

It is possible to combine the CM and RL models to improve coding performance and computational efficiency. For small values of K, the RL model tends to have similar efficiency as the CM model, but a much lower complexity due to the smaller number of symbols. For larger K values, the RL model tends to lose efficiency and becomes more complex. Because K is known in advance, the encoder can choose between CM and RL at run-time based on K. The decoder has access to the same information and can thus choose the same model as the encoder. It is even possible to use both models on the same vector, switching from from CM to RL once $K^{(n)}$ becomes smaller than an encoder-decoder-agreed threshold K_T.

3 Coding K

In some contexts, the value of K is agreed on between the encoder and decoder. If not, then we need to code K explicitly. In the proposed implementation, the pdf of K is adapted based on the data for $K < 15$. For $K \geq 15$, an exponentially-decaying distribution is assumed. The model is also conditioned on the expected value of K.

References

[1] J.-M. Valin, "Pyramid Vector Quantization for Video Coding", IETF draft, 2013. http://tools.ietf.org/html/draft-valin-videocodec-pvq-00

[2] T. R. Fischer, "A Pyramid Vector Quantizer", in *IEEE Trans. on Information Theory*, Vol. 32, 1986, pp. 568-583.

[3] J. P. Ashley, E. M. Cruz-Zeno, U. Mittal, and W. Peng, "Wideband Coding of Speech Using a Scalable Pulse Codebook," in *Proc. IEEE Workshop on Speech Coding*, 2000, pp. 148–15.

[4] Valin, J.-M., Vos. K., Terriberry, T.B., "Definition of the Opus codec", RFC 6716, Internet Engineering Task Force, 2012.