Logarithmic Cubic Vector Quantization

Christian ROHLFING

Institute of Communication Systems and Data Processing, RWTH Aachen University, 52074 Aachen, Germany

rohlfing@ind.rwth-aachen.de

Abstract. *In this paper we present "Logarithmic Cubic Vector Quantization" (LCVQ), a novel type of gain-shape vector quantization (GSVQ). In LCVQ, the vector to be quantized is decomposed into a gain factor and a shape vector which is a normalized version of the input vector. Both components are quantized independently and transmitted to the decoder.*
Compared to other GSVQ approaches, in LCVQ the input vectors are normalized such that all shape vectors are located on the surface of the unit hypercube. As a conclusion, the shape vector quantizer can be realized based on uniform scalar quantizers. This yields low computational complexity as well as high memory efficiency even in case of very high vector dimensions.
In order to demonstrate the coding efficiency of the proposed quantization scheme, LCVQ is compared to existing quantization schemes, the recently proposed logarithmic spherical vector quantization (LSVQ), logarithmic scalar quantization (LSQ) and adaptive quantization backward (AQB).

Keywords

Vector Quantization, Gain-shape Vector Quantization, Logarithmic Quantization.

1. Introduction

The quantizer is in general the key element in compression schemes for lossy speech and audio coding. In the design of a quantizer in practice, a good trade-off between complexity and quantization performance has to be achieved. On the one hand, scalar quantizers can be realized with low complexity but have only a moderate quantization performance. On the other hand, vector quantization techniques show promising results close to the theoretically achievable performance but have larger implementation costs. Recently, a gain-shape vector quantization (GSVQ) technique, logarithmic spherical vector quantization (LSVQ), has been proposed which is a reasonably well-balanced solution to this problem [1]. Internally, it requires a vector codebook which is computed in an offline manner [2].

"Logarithmic Cubic Vector Quantization" (LCVQ) is a novel type of gain-shape vector quantization which offers some advantages over LSVQ: It operates with low complexity and high memory efficiency even in case of very high input vector dimensions. Therefore, it is flexible regarding the possibility to adjust the bit rate at runtime. It allows higher input vector dimensions than LSVQ, since no vector codebook is required.

In this paper, LCVQ shall be investigated and compared to other quantization schemes: A generalization of gain-shape vector quantization is given in Section 2 as well as brief descriptions of logarithmic scalar quantization (LSQ), LSVQ and the novel LCVQ.
Section 3 deals with the quantization scheme adaptive quantization backward (AQB). It shows interesting similarities to the previously mentioned GSVQ techniques LSVQ and LCVQ.
The quantization performance of all quantization techniques are evaluated in Section 4. Finally, conclusions are given in Section 5.

2. Generalized Gain-shape Vector Quantization

In gain-shape vector quantization (GSVQ) [3], the input vector $\mathbf{x} \in \mathbb{R}^L$ with dimension L is decomposed into a gain factor $g \geq 0$ and a shape vector $\mathbf{c} \in \mathbb{R}^L$ which are then quantized independently by means of a scalar quantizer for g and a vector quantizer for $\mathbf{c}$ and transmitted to the decoder (refer to Figure 1).
The two components are computed as

$$g = \|\mathbf{x}\|_p = \left(\sum_{i=1}^{L} x_i^p \right)^{\frac{1}{p}} \qquad \text{and} \qquad \mathbf{c} = \frac{1}{g} \cdot \mathbf{x} \tag{1}$$

with $\|\mathbf{x}\|_p$ denoting the p-norm of a vector.

In the decoder, combining the quantized version of the gain factor, $\tilde{g} = \mathrm{Q_g}(g)$ and of the shape-vector, $\tilde{\mathbf{c}} = \mathrm{Q_c}(\mathbf{c})$ results in the overall reconstruction vector

$$\tilde{\mathbf{x}} = \tilde{g} \cdot \tilde{\mathbf{c}} \,. \tag{2}$$

The main difference between LSVQ and LCVQ can be demonstrated based on equation 1:

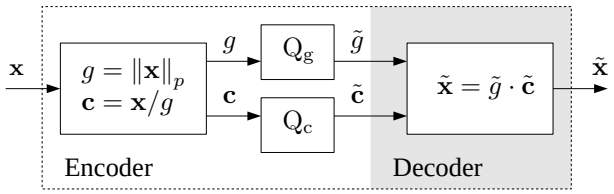


Fig. 1. Parallel quantization of the gain and shape component with $p = 2$ and $\mathrm{Q_c} = \mathrm{Q_{svq}}$ for LSVQ and $p = \infty$, $\mathrm{Q_c} = \mathrm{Q_{cvq}}$ for LCVQ.

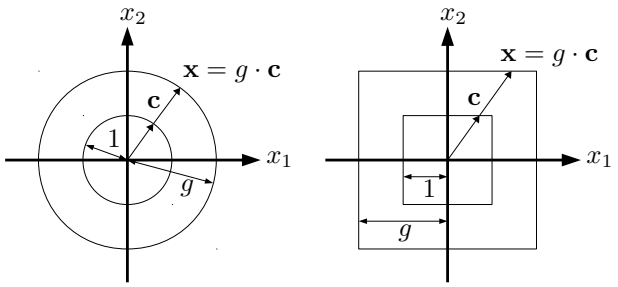
In LSVQ, setting $p = 2$ results in a normalization of input vector $\mathbf{x}$ to its Euclidean norm

$$g = \|\mathbf{x}\|_2 = \sqrt{\mathbf{x}^T \cdot \mathbf{x}}\,. \tag{3}$$

In contrast to this, LCVQ makes use of the uniform (maximum) norm with $p = \infty$ instead of the Euclidean norm:

$$g = \|\mathbf{x}\|_\infty = \max(|x_1|, \ldots, |x_L|)\,. \tag{4}$$

Using the Euclidean norm for normalization yields in the shape vectors $\mathbf{c}$ being located on the surface of the L-dimensional unit hypersphere. In contrast to this, normalizing with the unit norm results in $\mathbf{c}$ lying on the surface of the L-dimensional unit hypercube. The difference is illustrated in Figure 2.



(a) Euclidean norm (b) Uniform norm

Fig. 2. Comparison of $g = \|\mathbf{x}\|_2$ on the left and $g = \|\mathbf{x}\|_\infty$ on the right hand plot for vector dimension $L = 2$. The normalized vector $\mathbf{c} = \mathbf{x}/g$ lies on the surface of the unit sphere or unit cube, respectively.

2.1. Logarithmic Quantization of the Gain Factor

In LSVQ and LCVQ, the gain factor g is quantized using a logarithmic scalar quantizer (LSQ) $\mathrm{Q_g}$. Here, the A-law companding algorithm shall be used with the number of reconstruction levels N_g, compression factor A and stepsize Δg, as described in [4]. It can be shown, that in the case of LSQ, the stepsize scales with the reconstruction value $\tilde{g}$ [5]:

$$\Delta g(\tilde{g}) = \frac{1 + \ln(A)}{N_g} \cdot \tilde{g}\,. \tag{5}$$

The signal-to-noise ratio (SNR) of LSQ is independent of the input vector probability density function (PDF)

$$\begin{aligned}\mathrm{SNR_{lsq,A}}|_{\mathrm{dB}} &= 6.02 \cdot R + 10\log_{10}(3) \\ &\quad - 20\log_{10}(1 + \ln(A))\end{aligned} \tag{6}$$

with R denoting the bit rate of the quantizer. This independence comes at the price of a penalty-term of $10\log_{10}(3) - 20\log_{10}(1 + \ln(A))$ compared to the well-known 6 dB-per-bit-rule [5].

As LSQ can be considered as a special case of LSVQ and LCVQ for input vector dimension $L = 1$, it is used in Section 4 as a reference.

2.2. Logarithmic Spherical VQ (LSVQ)

In logarithmic spherical vector quantization, the Euclidean norm with $p = 2$ is used in equation 1 for normalization of input vector $\mathbf{x}$:

$$g = \|\mathbf{x}\|_2 = \sqrt{\mathbf{x}^T \cdot \mathbf{x}}\,. \tag{7}$$

In that case, the normalized shape vectors $\mathbf{c}$ are lying on the surface of the L-dimensional unit hypersphere.
The shape vector $\mathbf{c}$ is quantized using a spherical vector quantizer (SVQ) denoted by $\mathrm{Q_c} = \mathrm{Q_{svq}}$ whose codebook consists of N_{svq} spherical codevectors which are located on the surface of the unit sphere. The overall design goal of the codebook of $\mathrm{Q_{svq}}$ is to distribute the codevectors uniformly over the unit sphere surface.
[2] shows, that using LSQ for quantization of the gain factor $g = \|\mathbf{x}\|_2$ and SVQ for the shape vector $\mathbf{c}$ results in a SNR which is independent of the PDF of the input vector:

$$\begin{aligned}\mathrm{SNR}^{(I)}_{\mathrm{lsvq}} = \frac{N_{\mathrm{lsvq}}^{\frac{2}{L}}}{C_{\mathrm{lsvq}} \cdot \pi} \cdot \left(\frac{\Gamma\left(\frac{L}{2}\right)}{2 \cdot (1 + \ln(A))}\right)^{\frac{2}{L}} \\ \cdot \underbrace{\frac{\int\limits_{\mathbf{x} \in \mathbb{R}^L} p(\mathbf{x}) \cdot \|\mathbf{x}\|_2^2 \,\mathrm{d}\mathbf{x}}{\int\limits_{\tilde{\mathbf{x}} \in \mathbb{R}^L} p(\tilde{\mathbf{x}}) \cdot \|\tilde{\mathbf{x}}\|_2^2 \,\mathrm{d}\tilde{\mathbf{x}}}}_{\approx 1}\,.\end{aligned} \tag{8}$$

Equation 8 is a qualitative formula for the SNR which can be derived from the normalized quantizer point density function under high bit rate assumptions. C_{lsvq} denotes an unknown quantization cell form-factor and $\Gamma(z) = \int_0^\infty e^{-t} \cdot t^{z-1}\mathrm{d}t$ the gamma-function [6].

High bit rate approximations yield another more quantitative formula for the SNR in dB as

$$\begin{aligned}&\mathrm{SNR}^{(II)}_{\mathrm{lsvq}}|_{\mathrm{dB}} = 6.02 \cdot R_{\mathrm{eff}} \\ &- 10\log_{10}\left(\frac{L}{(L+1)^{\frac{L-1}{L}}} \cdot \left[2 \cdot \sqrt{\pi} \cdot \frac{\Gamma\left(\frac{L+1}{2}\right)}{\Gamma\left(\frac{L}{2}\right)}\right]^{\frac{2}{L}}\right. \\ &\left. \cdot \left[\frac{(1 + \ln(A))^2}{12}\right]^{\frac{1}{L}}\right)\,.\end{aligned} \tag{9}$$

This formula is used in Section 4 to compare the performance of LSVQ with the novel LCVQ scheme.

With N_g being the number of quantization reconstruction levels of the gain-component, the total number of LSVQ codevectors is

$$N_{\mathrm{lsvq}} = 2^{R_{\mathrm{eff}} \cdot L} = N_g \cdot N_{\mathrm{svq}} \tag{10}$$

with R_{eff} denoting the effective bit rate per vector dimension. Naturally, a bit allocation is needed to distribute the overall bit rate optimally over the gain and the shape quantizer's bit rates [2].

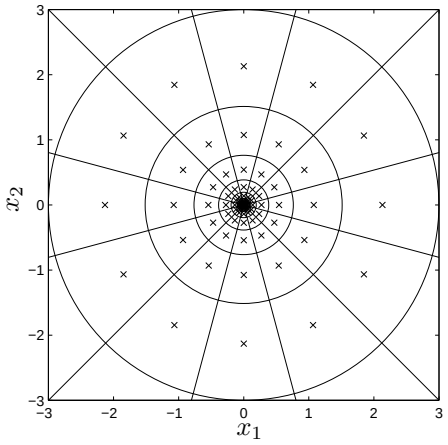


Fig. 3. LSVQ reconstruction values (denoted by crosses) and quantization cells for vector dimension $L = 2$ with number of spherical codevectors $N_{\mathrm{svq}} = 12$.

Figure 3 shows the overall LSVQ reconstruction values and quantization cells for the example of $L = 2$. The SVQ codevectors are located on versions of the shells of the unit sphere which are scaled by the LSQ reconstruction levels.

The design of spherical vector-codebooks used for $\mathrm{Q}_{\mathrm{svq}}$ is rather complex and shall not be discussed in this paper. Different procedures are given in [7] and [2].

2.3. Logarithmic Cubic VQ (LCVQ)

In "Logarithmic Cubic Vector Quantization", the input vector is normalized by its uniform norm

$$g = \|\mathbf{x}\|_{\infty} = \max(|x_1|, \ldots, |x_L|) = x_{l_0} \tag{11}$$

with $l_0 = \arg\max_l(|x_l|)$ denoting the index of the maximum value of $\mathbf{x}$ such that the normalized vector component $c_{l_0} = 1$.
As a result, the shape vectors $\mathbf{c}$ are lying on the surface of the L-dimensional unit hypercube with edge length equal to 2 as depicted in Figure 2(b).
The quantizer of the shape vector $\mathrm{Q_c} = \mathrm{Q}_{\mathrm{cvq}}$ can be realized in a very efficient way as it consists of $L - 1$ uniform scalar quantizers Q_{sq} for each dimension with index $l \neq l_0$

$$\tilde{c}_l = \begin{cases} \mathrm{Q}_{\mathrm{sq}}(c_l), & \text{if } l \neq l_0 \,, \\ 1, & \text{if } l = l_0 \,. \end{cases} \tag{12}$$

Each uniform scalar quantizer Q_{sq} has the same number of reconstruction levels N_c and the stepsize

$$\Delta c = \frac{2}{N_c} \,. \tag{13}$$

The total number of LCVQ reconstruction values is calculated as

$$N_{\mathrm{lcvq}} = 2^{R_{\mathrm{eff}} \cdot L} = N_g \cdot N_{\mathrm{cvq}} = N_g \cdot 2 \cdot L \cdot N_c^{L-1} \tag{14}$$

with N_{cvq} denoting the number of reconstruction values per cube shell. The factor $2 \cdot L$ gives the number of surfaces of a L-dimensional hypercube.

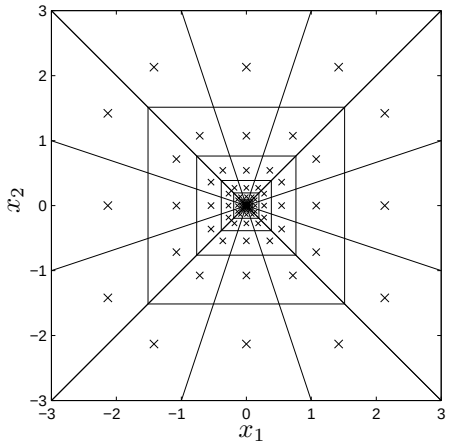


Fig. 4. LCVQ reconstruction values (denoted by crosses) and quantization cells for vector dimension $L = 2$ with number of quantization cells per dimension $N_c = 3$.

The overall LCVQ reconstruction values and quantization cells for vector dimension $L = 2$ are depicted in Figure 4. The reconstruction values are lying on cube shells which are scaled by the LSQ reconstruction levels and are equally distributed over the hypercube surface (similar to the SVQ codevectors, refer to Figure 3).

Using only scalar quantizers, it becomes clear that LCVQ offers the previously mentioned advantages such as low memory and implementation cost. Note that LCVQ has no limits regarding the input vector dimension in comparison to LSVQ, where the vector dimension is bounded by the codebook design.

3. Adaptive Quantization Backward (AQB)

In adaptive quantization backward, which was proposed in [9], the quantization stepsize Δx is dynamically

adapted to the instantaneous signal power $\tilde{\sigma}_x$ of the quantized input signal $\tilde{x}(k)$ with k being the current time index. $\tilde{\sigma}_x$ is estimated recursively using the latest transmitted value $Z(k-1)$:

$$\frac{\Delta x(k)}{\Delta x(k-1)} = \frac{\tilde{\sigma}_x(k)}{\tilde{\sigma}_x(k-1)} = f\{Z(k-1)\} = M(k-1)\,. \tag{15}$$

$M(k)$ denotes the so-called stepsize multiplier which is a function of $Z(k)$. $M(k)$ can be precalculated and stored in a table.

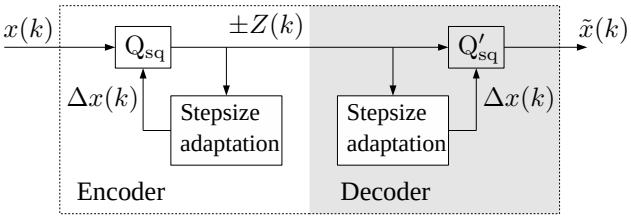


Fig. 5. Block diagram of the adaptive quantization backward scheme.

Figure 5 shows a block diagram of the AQB method: $\mathrm{Q_{sq}}$ is a common uniform scalar quantizer with number of reconstruction levels $N_{\mathrm{aqb}} = 2^R$ and stepsize $\Delta x(k)$. $\mathrm{Q_{sq}}$ provides $Z(k)$ as an intermediate value at time index k which is also transmitted to the decoder. In the decoder, $Z(k)$ is transformed by $\mathrm{Q'_{sq}}$ to the quantization reconstruction level $\tilde{x}(k)$. The stepsize-adaptation block is the same in encoder and decoder and uses equation 15 to estimate the instantaneous stepsize $\Delta x(k)$.

The AQB shall be considered as a reference quantizer in Section 4 as it shows similarities to LCVQ and LSVQ in the sense that all these algorithms have in common to estimate the instantaneous signal variance: AQB uses a direct estimation of the variance, whereas LCVQ and LSVQ are adapting to the variance by normalization of the input vector to the gain factor.

4. Evaluation

For the evaluation, a large number of signal vectors was generated following a normal or uniform distribution. These vectors were quantized afterwards with an LCVQ or an AQB quantizer for different vector dimensions $L \in \{2, 8, 48\}$. Finally, the signal-to-quantization-noise ratio (SNR) was calculated for each quantization scheme. The optimal LCVQ bit allocation for N_g and N_c was determined experimentally. The simulated SNRs of LCVQ and AQB are compared to the theoretical SNRs of LSVQ (equation 9) and LSQ (equation 6) for different bit rates $R_{\mathrm{eff}} \in [1.5, 5]$.

The companding factor A is set to 5000 for all logarithmic quantizers, which was found in [10] as a reasonable trade-off between dynamic range and performance for audio signals.

Figure 6 shows the results of LCVQ compared to LSVQ, AQB and LSQ for uniformly and Figure 7 for normally distributed input vectors. LCVQ achieves almost the same SNR as LSVQ for the uniform input vector distribution for all input vector dimensions L. LCVQ shows better results than AQB for high bit rates and above vector-dimension $L = 8$.

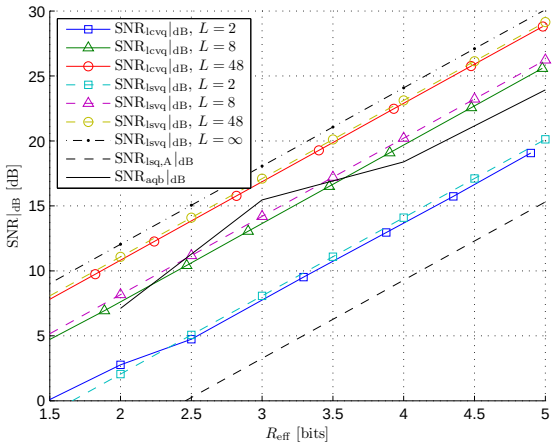


Fig. 6. SNR for uniformly distributed input vectors $\mathbf{x}$.

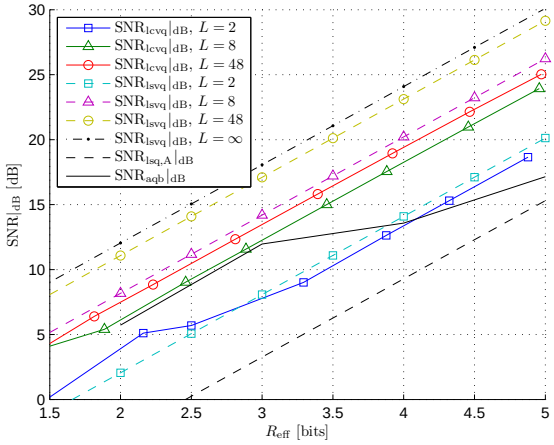


Fig. 7. SNR for input vectors $\mathbf{x}$ following a normal distribution with mean $\mu_x = 0$ and variance $\sigma_x^2 = 1$.

For normally distributed input vectors, LSVQ outperforms LCVQ for all vector-dimensions L, whereby the difference between the LSVQ and LCVQ SNR increases with L. For vector-dimension $L = 8$ and lower bit rates, LCVQ provides a similar performance compared to AQB and performs better for high bit rates.
The worst SNR results in both PDF scenarios are obtained by LSQ which is the special case of LSVQ and LCVQ for vector dimension $L = 1$.

LCVQ quantization cells introduce a larger quantization error due to their square shape compared to the non-square shaped LSVQ cells [8]. The overall square configuration of LCVQ reconstruction values is not optimal for spherically distributed sources either [2].

In practice, it is not feasible to generate LSVQ codevectors above a vector dimension of $L = 16$ whereas LCVQ is still able to quantize vectors with very large dimensions. This is one of the key advantages of LCVQ compared to LSVQ.

5. Conclusions

In this paper, a novel gain-shape vector quantization technique, "Logarithmic Cubic Vector Quantization", was proposed and compared to other quantization schemes, particularly to logarithmic spherical vector quantization. LCVQ offers the advantages of low complexity, high memory efficiency and adaptivity to different bit rates at runtime. LSVQ shows slightly better SNR results than LCVQ. By a significant increase of the input vector dimension, which is not possible for LSVQ, LCVQ can overcome the performance loss compared to LSVQ.

Acknowledgements

Research described in the paper was supervised by Dr.-Ing. H. Krüger and Prof. Dr.-Ing. P. Vary from the Institute of Communication Systems and Data Processing, RWTH Aachen University.

References

[1] KRÜGER, H., GEISER, B., VARY, P.; LI, H. T., ZHANG, D. *Gosset lattice spherical vector quantization with low complexity*. 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2011.

[2] KRÜGER, H. *Low delay audio coding based on logarithmic spherical vector quantization*. Aachener Beiträge zu Digitalen Nachrichtensystemen (ABDN), Verlag Mainz, 2010.

[3] SABIN, M., GRAY, R., *Product code vector quantizers for waveform and voice coding*. IEEE Transactions on Acoustics, Speech and Signal Processing, 1984.

[4] JAYANT, N. S., & NOLL, P. *Digital Coding of Waveforms*. Prentice Hall Signal Processing Series. Prentice-Hall, 1984.

[5] VARY, P., & MARTIN, R. *Digital speech transmission: enhancement, coding and error concealment*. Journal of the Acoustical Society of America (Vol. 121. Wiley & Sons, 2006.

[6] BRONSTEIN, I. N. $ SEMENDJAJEW, K. A. *Taschenbuch der Mathematik*. Teubner Verlag, 1991.

[7] KRÜGER, H. & VARY, P. *SCELP: Low Delay Audio Coding with Noise Shaping Based on Spherical Vector Quantization*. Proceedings of European Signal Processing Conference (EUSIPCO), EURASIP, 2006.

[8] GERSHO, A. *Asymptotically Optimal Block Quantization*. IEEE Transactions on Information Theory, 1979.

[9] JAYANT, N. S. *Adaptive Quantization with a One-Word Memory*. Journal of the Acoustical Society of America, Volume 54, Acoustical Society of America, 1973.

[10] KRÜGER, H., SCHREIBER, R., GEISER, B. & VARY, P. *On Logarithmic Spherical Vector Quantization*. Proceedings of International Symposium on Information Theory and its Applications (ISITA), Society of Information Theory and its Applications (SITA), 2008.

About Authors...

Christian ROHLFING was born in Krefeld, Germany. He began his studies of Computer Engineering at RWTH Aachen University in 2006 and is currently writing his diploma thesis at the Institute of Communication Systems and Data Processing, RWTH Aachen University.