CODING OF PREDICTION RESIDUAL IN MPEG-4 STANDARD FOR LOSSLESS AUDIO CODING (MPEG-4 ALS)

Yuriy A. Reznik

RealNetworks, Inc.
2601 Elliott Avenue, Suite 1000
Seattle, WA 98121

ABSTRACT

We describe two alternative schemes for encoding of prediction residual adopted in the MPEG-4 ALS standard for lossless audio coding. We explain choices of algorithms used in their design and provide both analytical and experimental analysis of their performance.

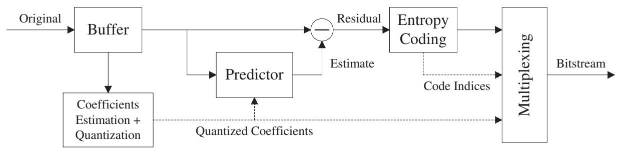

Fig. 1. MPEG-4 ALS encoder.

1. INTRODUCTION

Working draft of an amendment ISO/IEC 14496-3:2001/AMD 4 (MPEG-4 ALS) [8] represents the latest planned addition to a suite of MPEG Audio standards [5], defining technology for lossless coding of PCM audio signals.

In a nutshell, MPEG-4 ALS is a forward-adaptive Linear Predictive Coder (LPC) in which predicted signal is quantized to the resolution of the input PCM signal. Combined with lossless compression and transmission of quantized filter coefficients and the residual this insures lossless reconstruction of the original signal.

The structure of this algorithm is shown in Fig.1. It contains all standard building blocks of an LPC coder: buffer for storing blocks of input PCM signal, prediction filter, module for estimation and quantization of filter coefficients, entropy coding and multiplexing units. Specific features of MPEG-4 ALS encoder include its ability to change order of the predictor, use different block sizes, inject headers insuring random access to compressed data, etc.

In this paper we will describe two alternative techniques used in this algorithm for encoding of prediction residual. The first scheme, provided in the first reference model of MPEG-4 ALS [8] is based on the use of simple parametric Golomb-Rice codes [4, 16]. The other scheme, accepted as a first core experiment [14] is more complicated, and it uses several coding techniques, such as block Gilbert-Moore, fixed-length, and Golomb-Rice codes.

In describing these schemes our main goal will be to explain the choices of coding algorithms and parameters used in their design. We will complement our exposition by presenting experimental results.

2. RICE CODING OF PREDICTION RESIDUAL

Golomb codes [4] represent a special case of Huffman codes constructed for sources with geometric distribution of symbols: $\Pr\{r_\theta = i\} = (1-\theta)\,\theta^i$ where $\theta \in (0,1)$. The code $G_m(i)$ consists of a series of k ones, where k is the result of a division $k = \lfloor i/m \rfloor$, followed by a 0-bit, and a $\lceil \log_2 m \rceil$-bit representation of the remainder $i \mod m$. It has been shown (see [4, 2]) that an optimal for a source r_θ parameter m can be calculated as follows:

$$m = \lfloor -\log(1+\theta) \,/\, \log\theta \rfloor \,. \tag{1}$$

If this quantity is further rounded to the nearest power of 2, then the divisions can be replaced by shifts, resulting in an extremely simple and fast encoding. The codes $GR_s(i) := G_{2^s}(i)$ are often called Golomb-Rice or Rice codes [16] with parameter s.

Robinson [19] has already observed that the distribution of the residual signal in LPC-based audio encoders can be closely modelled by a Laplacian (or two-sided geometric) distribution. This means, that in order to apply Golomb-Rice codes one simply needs to flip the negative side of the distribution and merge it with the positive one. In MPEG-4 ALS this is accomplished by a mapping:

$$r_i^+ = \left[\begin{array}{ll} 2\,r_i & \text{if } r_i \geqslant 0\,, \\ -2\,r_i - 1 & \text{if } r_i < 0\,. \end{array} \right. \tag{2}$$

In order to estimate the optimal Golomb-Rice parameter s for a block of residuals $r_1, \ldots, r_n$, the reference implementation of MPEG-4 ALS calculates their absolute mean:

$$\mu_n = \frac{1}{n} \sum_{i=1}^{n} |r_i| \;, \tag{3}$$

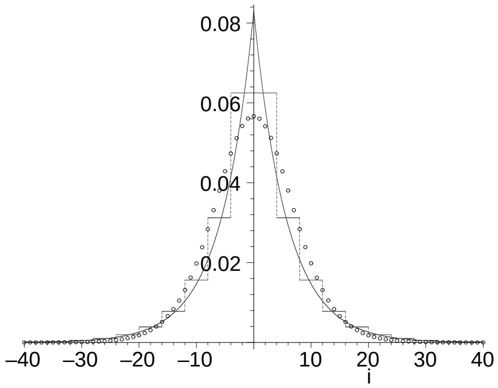

Fig. 2. Observed residual distribution (circles), model Laplacian distribution (solid line), and the distribution corresponding to the lengths of Golomb-Rice codes (dashed line).

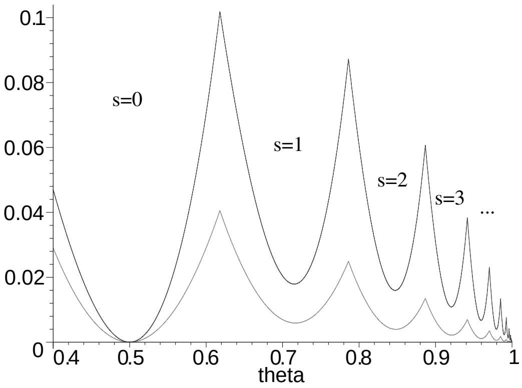

Fig. 3. Redundancy $R^{GR}(x_\theta) = E\{|GR_s(x_\theta)|\} - H(x_\theta)$ and normalized redundancy $R^{GR}(x_\theta)/H(x_\theta)$ of Golomb-Rice codes for a geometric source x_θ: $\Pr\{x_\theta = i\} = (1-\theta)\,\theta^i$.

and then uses:

$$s = \lfloor \log_2 \mu_n + C_1 \rfloor \ , \qquad (4)$$

where $C_1 \approx 0.97$ is a constant.

This estimator has already been used in compression algorithms such as SHORTEN [19] or LOCO-1 [24], and it is based on the fact that sample absolute mean μ_n converges (with large n) to the first absolute moment $E\{|r|\}$ of the distribution. Thus, in a Laplacian model: $E\{|r|\} = \int_{-\infty}^{\infty} \Pr\{r=\rho\}\,|\rho|\,d\rho = \int_{-\infty}^{\infty} \frac{1}{\sqrt{2}\lambda}\, e^{-\frac{\sqrt{2}}{\lambda}|\rho|}\,|\rho|\,d\rho = \frac{1}{\sqrt{2}}\lambda$, where λ is the variance. In turn, λ can be converted in a parameter θ of the quantized geometric distribution (see Appendix A), and then by (1), setting $s = \log_2 m$, and ignoring smaller than $O(1)$ terms we arrive at (4).

Obtained in such a way code parameter s is transmitted along with the encoded residuals $GR_s\left(r_1^+\right), \ldots, GR_s\left(r_n^+\right)$. To facilitate a higher degree of adaptation during transients in audio signals MPEG-4 ALS includes a mode in which each block is divided in several smaller sub-blocks encoded using different (individually estimated for each sub-block) code parameters.

In Fig.2 we show a typical observed distribution of the residual, its approximation by a Laplacian distribution, and the distribution corresponding to the lengths of the resulting Golomb-Rice codes. It is clear there is a divergence between the observed distribution and one that is being encoded. However, after an extensive experimental study [13] using MPEG audio sequences [6] we have found that the estimated average redundancy of Golomb-Rice encoding represents only $1.6452..\%$ of the bitrate occupied by the residuals. In other words, given the simplicity of this coding scheme, it works remarkably well in this application.

3. IMPROVED RESIDUAL CODING

The main motivations for including another residual coding scheme in MPEG-4 ALS were: a) the requirement to achieve a better compression than what is claimed by today's state of the art algorithms [6], b) realization of the fact that the efficiency of encoding of residuals is critical for the performance of the entire algorithm, and c) the fact that Golomb-Rice codes have fundamental performance limits (see [2, 23], also Fig.3), hence further progress cannot be achieved without using a more efficient technique.

On the other hand, looking for alternative schemes we also had to consider their complexity, so our final goal was to design an algorithm that substantially reduces the redundancy of encoding of prediction residuals while making the whole encoding/decoding process only slightly more complex [14].

In order to achieve this goal, the following techniques have been incorporated.

3.1. Higher-resolution representation of parameter s.

Similar to the previous scheme, at the beginning of each block we transmit parameter s describing the probability distribution of the residuals. We use the same technique for estimation of this parameter, but we quantize and transmit it using higher precision.

3.2. Partition of the residual distribution.

To restrict memory usage we only construct high-efficiency block codes for a central region $[-r_{\max}, r_{\max}]$ of the residual distribution (see Fig.4). The tails are still encoded using Golomb-Rice codes as described in the previous section.

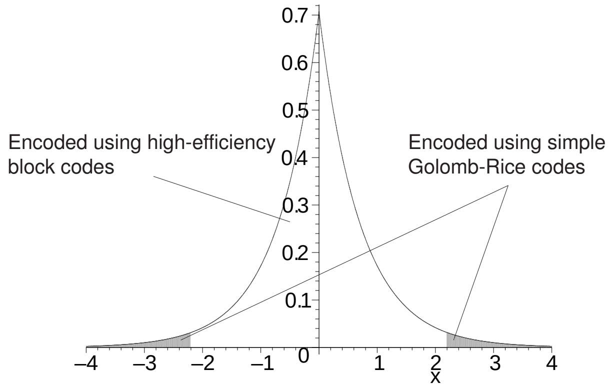

Fig. 4. Partition of the residual distribution.

3.3. The use of block Gilbert-Moore codes for nested ordered distributions.

To encode residual values in the central region we have chosen to use block Gilbert-Moore codes [3]. The key advantages of these codes[1] come from the facts that:

- their average redundancy rate is *decreasing with the length of an encoded block* (cf. [3, 9, 23]):

$$R^{GM}(n) = \frac{3/2+\delta(n)}{n} < \frac{2}{n}, \tag{5}$$

  where n is the block length, and $\delta(n)$ is an oscillating function of bounded magnitude $|\delta(n)| < 1/2$.

- they can be approximately constructed using memory-efficient arithmetic encoders [17, 12, 18, 25]. The added redundancy of such an implementation with m-symbols alphabet, t-bit code registers, and τ-bit probability representations is bounded by [22]:

$$R^{AC}(t,\tau) < m\,(\tau + \log e)\; 2^{-(t-2)} \tag{6}$$

  and can be easily controlled by properly selecting t and τ.

- the construction of these codes is based on cumulative probabilities of symbols, making it possible not only to encode a source with a given distribution, but also *all reduced-resolution versions of this source*.

For example, given symbols $a_1, \ldots, a_m$ with non-increasing probabilities $p_1 \geqslant \ldots \geqslant p_m$, the code construction requires a table with m quantities $s_i = \sum_{j=i}^{m} p_j$. If we now quantize this source, for example, by skipping the last bit, then the new symbols $\hat{a}_1, \ldots, \hat{a}_{\lfloor m/2 \rfloor}$ will have (also non-increasing) probabilities $\hat{p}_i = p_{2i} + p_{2i+1}$. It is clear, that the new cumulative probabilities $\hat{s}_i = s_{2i}$ simply represent a *sub-set of the same table* $s_1, \ldots, s_m$. As a general rule, if Δ represents the number of skipped bits, then the encoding of such symbols can be accomplished by using probabilities $\hat{s}_i = s_{i\,2^{\Delta}}$.

[1] Sometimes these codes are also called Elias-Shannon-Fano codes [1].

3.4. Block-size adaptive reduction of the alphabet size.

In order to speed up both encoding and decoding and to further reduce memory usage we split and transmit k least significant bits (LSBs) of the residuals in central region directly[2]. The number of directly transmitted bits k is selected using:

$$k = \left[\begin{array}{ll} 0, & \text{if } s \leqslant B\,, \\ s - B, & \text{if } s > B\,, \end{array} \right. \tag{7}$$

where s is the code parameter (4), and B is a quantity depending on the block size n:

$$B = \lfloor (\log n - 3)\,/2 \rfloor \;. \tag{8}$$

The associated parameter Δ (the number of missing bits) used in our Gilbert-Moore encoder is obtained using:

$$\Delta = 5 - s + k\,, \tag{9}$$

where 5 is the maximum value of B, given that MPEG4 ALS's block size $n \leqslant 2^{13}$.

In Appendix A we will show that the described choice of the parameter k limits the redundancy due to direct transmission of LSBs to approximately 25% of the total redundancy of residual coding. Choices of other parameters, such as $r_{\max}$, t, and τ, are also based on similar constraints and will be further explained in Appendix A.

Overall, our implementation of the described coding scheme uses only 2K words of memory to store probability tables, and involves 2 multiplications per encoded or decoded sample. Given the fact that the order of the prediction filter in MPEG-4 ALS typically fluctuates in the range of $8 - 16$, these two extra multiplications should have only minor $(12.5 - 25\%)$ effect on the overall performance.

4. EXPERIMENTAL RESULTS

The results of our experimental study are summarized in Table 1. The test material was taken from the standard audio sequences for MPEG-4 Lossless Coding [6]. It comprises nearly 1 GB of stereo waveform data with sampling rates of 48, 96, and 192 kHz, and resolutions of 16, 20, and 24 bits.

The column "compression" in Table 1. contains ratios of compressed file sizes to the sizes of original waveform files, expressed in $\%$s. Both compression and decompression speeds are measured in seconds. To run these tests we used a PIII-M computer with 500M bytes of RAM.

In these tests we used reference-model implementation of the MPEG-4 ALS encoder [11].

[2] This idea is similar to a more elaborate "alphabet grouping" technique [21], but our approach is much simpler because we use groups of equal size.

Format	Compression		Enc. Time		Dec. Time	
	RM0	*CE1*	*RM0*	*CE1*	*RM0*	*CE1*
48kHz-16	46.8	46.3	44.7	55.8	17.3	24.5
48kHz-20	64.2	63.9	60.9	58.5	25.6	26.3
48kHz-24	64.2	63.9	62.4	58.5	24.8	26.3
96kHz-16	31.3	30.7	77.4	104.4	31.9	44.8
96kHz-20	50.2	49.9	114.9	117.0	46.9	52.2
96kHz-24	48.8	48.4	109.7	116.5	47.5	52.0
192kHz-16	22.3	21.6	59.3	76.1	24.9	33.3
192kHz-20	42.3	41.9	88.7	100.2	50.7	53.2
192kHz-24	39.5	39.0	91.3	99.7	47.9	51.8
Total	**45.2**	**44.7**	**709.3**	**786.8**	**317.4**	**364.2**

Table 1. Performance analysis for different audio formats. Algorithm *RM0* uses Golomb-Rice codes, while algorithm *CE1* uses improved residual coding scheme.

5. REFERENCES

[1] T. M. Cover and J. M. Thomas, *Elements of Information Theory*, (John Wiley & Sons, New York, 1991).

[2] R. G. Gallager, and D.C. Van Voorhis, Optimal Source Codes for Geometrically Distributed Integer Alphabets, *IEEE Trans. Inform. Theory*, **21** (3) (1975) 228–230.

[3] E. N. Gilbert and E. F. Moore, Variable-Length Binary Encodings, *Bell Syst. Tech. J.*, 7 (1959) 932–967.

[4] S. W. Golomb, Run-Length Encodings, *IEEE Trans. Inform. Theory*, **12** (7) (1966) 399-401.

[5] ISO/IEC 14496-3:2001, *Information technology - Coding of audio-visual objects - Part 3: Audio* (International Standard, 2001).

[6] ISO/IEC JTC1 /SC29/WG11 N5040, *Call for Proposals on MPEG-4 Lossless Audio Coding* (Klagenfurt, AT, July 2002)

[7] ISO/IEC JTC1/SC29/WG11 N5718, *Working Draft of ISO/IEC 14496-3:2001/AMD 4, Audio Lossless Coding* (65th MPEG Meeting, Trondheim, Norway, July 2003).

[8] ISO/IEC JTC1/SC29/WG11 N6012, *Working Draft 1.0 of ISO/IEC 14496-3:2001/AMD 4, Audio Lossless Coding* (66th MPEG Meeting, Brisbane, Australia, October 2003).

[9] R. E. Krichevsky, Universal Data Compression and Retrieval, (Kluwer, Norwell, MA, 1993).

[10] T. Liebchen, *Detailed technical description of the TUB 'lossless only' proposal for MPEG-4 Audio Lossless Coding*, ISO/IEC JTC1/SC29/WG11 M9781 (64th MPEG Meeting, Awaji island, Japan 2003).

[11] T. Liebchen at al., *MPEG4 ALS source code*, ftp://ftlabsrv.nue.tu-berlin.de/mp4lossless

[12] R. Pasco, *Source coding algorithm for fast data compression* (Ph.D. thesis, Stanford University, 1976).

[13] Yu. A. Reznik, *Performance Limits of Codes for Prediction Residual*, ISO/IEC JTC1/SC29/WG11 M9890, (65th MPEG Meeting, Trondheim, Norway, July 2003).

[14] Yu. A. Reznik, *Proposed Core Experiment for Improving Coding of Prediction Residual in MPEG-4 ALS RM0*, ISO/IEC JTC1/SC29/WG11 M9893, (65th MPEG Meeting, Trondheim, Norway, July 2003).

[15] Yu. A. Reznik and W. Szpankowski, Asymptotic Average Redundancy of Adaptive Block Codes, *2003 IEEE Intl. Symp. Inform. Theory* (Yokohama, Japan, 2003).

[16] R. F. Rice, *Some practical universal noiseless coding techniques - Parts I-III*, JPL Tech. Reps. JPL-79-22 (1979), JPL-83-17 (1983), and JPL-91-3 (1991).

[17] J. J. Rissanen, Generalized Kraft inequality and arithmetic coding, *IBM J. Res. Dev.*, 20 (1976) 198–203.

[18] J. J. Rissanen and G. G. Langdon, Arithmetic coding, *IBM J. Res. Dev.*, 23 (2) (1979) 149–162.

[19] T. Robinson, *SHORTEN: Simple Lossless and Near-Lossless Waveform Compression*, Tech. Rep. CUED/F-INFENG/TR.156 (Cambridge University, UK, 1994).

[20] F. Rubin, Arithmetic stream coding using fixed precision registers, *IEEE Trans. Inform. Theory*, 25 (6) (1979) 672–675.

[21] B. Ya. Ryabko and J. Astola, Fast Codes for Large Alphabet Sources and its Application to Block Encoding, *2003 IEEE Intl. Symp. Inform. Theory* (Yokohama, Japan, 2003).

[22] B. Ya. Ryabko and A. N. Fionov, Efficient Method of Adaptive Arithmetic Coding for Sources with Large Alphabets, *Probl. Inform. Transm.* **35** (1999) 95–108 (in Russian).

[23] W. Szpankowski, Asymptotic Average Redundancy of Huffman (and Other) Block Codes, *IEEE Trans. Information Theory*, **46** (7) (2000) 2434–2443.

[24] M. J. Weinberger, G. Seroussi, and G. Sapiro, LOCO-1: A Low Complexity, Context-Based, Lossless Image Compression Algorithm, *Proc. 1996 Data Compression Conference*, (Snowbird, UT 1996) 140–149.

[25] I. H. Witten, R. Neal, J.G. Cleary, Arithmetic Coding for Data Compression, *Comm. ACM.*, 30 (6) (1987) 520–541.

A. SUPPORTING MATERIAL – WILL NOT APPEAR IN THE FINAL ABSTRACT!

A.1. Model and Entropy Rate of the Residual Signal

Let's assume that the residual signal has a Laplacian distribution:

$$\Pr\{r_\lambda = \rho\} = \frac{1}{\sqrt{2}\lambda}\, e^{-\frac{\sqrt{2}}{\lambda}|\rho|}\,, \qquad (10)$$

where λ is the variance.

In order to achieve lossless reconstruction MPEG-4 ALS quantizes the residual[3] r_λ to the resolution of the original PCM signal using the standard uniform mid-thread quantizer. Indeed, after quantization we are dealing with a discrete signal $r_\lambda^{[w]}$ with probabilities:

$$\begin{aligned}\Pr\{r_\lambda^{[w]} = i\} &= \int_{(i-\frac{1}{2})2^{-w}}^{(i+\frac{1}{2})2^{-w}} \Pr\{r_\lambda = \rho\}\, d\rho \\ &= \left[\begin{array}{ll} 1-\sqrt{\theta} & \text{if } i=0\,, \\ \frac{1}{2}\left(\frac{1}{\sqrt{\theta}}-\sqrt{\theta}\right)\theta^{|i|} & \text{if } |i|>0\,, \end{array}\right. \qquad (11)\end{aligned}$$

where

$$\theta = e^{-\frac{\sqrt{2}}{\lambda\, 2^w}}\,, \qquad (12)$$

and w is a PCM word length.

For further convenience, we will introduce a variable

$$\Lambda = \lambda\, 2^w\,, \qquad (13)$$

using which, we can estimate the entropy rate of the quantized signal $r_\lambda^{[w]}$ as follows:

$$\begin{aligned}H\left(r_\lambda^{[w]}\right) &= -\sum_{i=-\infty}^{\infty} \Pr\{r_\lambda^{[w]} = i\} \log \Pr\{r_\lambda^{[w]} = i\} \\ &= -\frac{1}{2}\frac{\sqrt{\theta}\,(1+\theta)}{1-\theta}\log\theta - \sqrt{\theta}\,\log(1-\theta) \\ &\quad -\left(1-\sqrt{\theta}\right)\log\left(1-\sqrt{\theta}\right) + \sqrt{\theta} \\ &= \log\Lambda + \log\left(\sqrt{2}e\right) + \frac{\log e}{12\,\Lambda^2} + O\left(\frac{1}{\Lambda^3}\right) \qquad (14)\end{aligned}$$

Obtained expression (14) confirms that $H\left(r_\lambda^{[w]}\right) \approx w + H(r_\lambda)$, where $H(r_\lambda) = \log\lambda + \log\left(\sqrt{2}e\right)$ is the differential entropy of the original signal (10) (see, e.g. [1]), but more importantly, it gives us an explicit formula for a subsequent term, which we will allow us to estimate redundancy due to subsequent requantization of this signal.

A.2. Redundancy due to Direct Transmission of LSBs

From (14) we already know how to estimate the entropy rate of the quantized Laplacian signal $r_\lambda^{[w]}$. This also gives us an estimate for the minimum rate of encoding of $r_\lambda^{[w]}$.

If instead of $r_\lambda^{[w]}$ we will choose to encode its $w-k$ most significant bits and transmit the remaining k bits directly, then the minimum achievable rate of such an encoding can be estimated as $H(r_\lambda^{[w-k]}) + k$. Consequently, the redundancy due to direct transmission of k LSBs:

$$\begin{aligned}R\left(r_\lambda^{[w]}, k\right) &\geqslant H\left(r_\lambda^{[w-k]}\right) + k - H\left(r_\lambda^{[w]}\right) \\ &= \frac{\log e}{12}\frac{2^{2\,k}}{\Lambda^2}\left[1-2^{-2\,k}\right] + O\left(\frac{1}{\Lambda^3}\right) \qquad (15)\end{aligned}$$

If we now need to construct a code such that its acceptable relative redundancy $\frac{R(r_\lambda^{[w]},k)}{H\left(r_\lambda^{[w]}\right)} \leqslant \delta$, where δ is a given parameter, then based on (15) we can choose k:

$$k(\delta) \leqslant \log\Lambda + \frac{1}{2}\log\delta + \frac{1}{2}\log(12\ln 2) + O\left(\frac{1}{\Lambda^2\delta}\right)\,.$$

A.3. Redundancy due to Golomb-Rice Encoding of Tails

We want to set tail points $\pm r_{\max}$ such that the combined redundancy rate is only slightly (within 25%) worse than the redundancy $R(n)$ of codes in the central region:

$$\Pr\left\{\left|r_\lambda^{[w]}\right| \leqslant r_{\max}\right\} R(n) + \Pr\left\{\left|r_\lambda^{[w]}\right| > r_{\max}\right\} R_{\max}^{GR} \leqslant 1.25\, R(n)\,, \qquad (16)$$

where n is the block size, and $R_{\max}^{GR} \approx R^{GR}(0.618..) = 0.10193..$ is the maximum (per symbol) redundancy of Golomb-Rice codes (see Fig.3).

From (11) we obtain:

$$\Pr\left\{\left|r_\lambda^{[w]}\right| > r_{\max}\right\} = \frac{\theta^{r_{\max}}}{\sqrt{\theta}} = e^{-\frac{2r_{\max}-1}{\sqrt{2}\,\Lambda}}\,, \qquad (17)$$

and by setting $R^C(n) = 2/n$ we arrive at:

$$r_{\max} = \frac{\ln 2}{\sqrt{2}}\left(\log n + \log\frac{1}{2\,R_{\max}^{GR}}\right)\Lambda + O\left(\frac{\Lambda}{n}\right)\,. \qquad (18)$$

[3] More precisely MPEG-4 ALS quantizes the predicted value before subtracting it from the original, but doing it in an inverse order will obviously produce the same result.