Optimizing Towards a Globally Consistent Metric in a Practical AV1 Encoder

Timothy B. Terriberry

July 22, 2020

1 Introduction

The purpose of this work is to define a single, global metric for a practical video encoder and show how the various sub-problems in the encoder can be arranged to consistently optimize for that metric. The basic idea of rate-distortion optimization is well established, and the vast majority of video encoding implementations rely heavily on its principles. However, there are many details that must be considered by a practical encoder implementation that are not as well established.

We take the AV1 format as our primary target for the encoder. Although some of the discussion will be AV1-specific, many of the principles derived apply equally well to other formats. Furthermore, we take the Rust AV1 encoder `rav1e` as our target implementation. While still in development and not feature complete, it contains all of the relevant pieces needed for the discussion at hand. These include things such as the allocation of bits between color planes, adjustments to account for human perception, boosts applied to certain frames to improve their usefulness for prediction, and other details specific to AV1.

Many real-world encoder implementations do not handle these details in a consistent way. Instead, the underlying metric that the individual pieces of the encoder optimize towards varies depending on the problem being solved. As a result, the different pieces fight with each other, leading to an overall sub-optimal result. Unfortunately, once established, such discrepancies become very hard to displace, as the encoder becomes tuned until the design of the encoder itself is trapped in a local minimum. Improvements to one piece of the encoder can lead to overall worse results, as they make that portion more competitive with the other pieces with which it is in contention. This makes it more difficult to advance the development of such an encoder.

2 The Metric

To define our global metric, we start with a very simple expression of the rate-distortion cost:

$$C_{\text{video}} \triangleq D + \lambda R \ . \tag{1}$$

This is the cost our encoder will attempt to minimize over the set of possible encoded bitstreams. Here, R is the total size of the video in bits, and D is a *block-wise weighted* squared error:

$$D \triangleq \sum_{f=0}^{F-1} \sum_{p=0}^{P-1} \sum_{i=0}^{\left\lceil \frac{W_p}{8} \right\rceil - 1} \sum_{j=0}^{\left\lceil \frac{H_p}{8} \right\rceil - 1} \sum_{x=8i}^{\mathfrak{r}_{p,i}} \sum_{y=8j}^{\mathfrak{b}_{p,j}} w_{f,p,i,j} |I(f,p,x,y) - \hat{I}(f,p,x,y)|^2 \tag{2}$$

Here, f indexes the frames, F is the total number of frames, and p indexes the color planes in each frame, of which there are P per frame. Similarly, W_p and H_p are the width and height of the p^{th} plane in each frame, indexed by x and y. I corresponds to the input image sequence to be encoded, and $\hat{I}$ corresponds to the encoded image sequence. Each 8×8 block of pixels is weighted by a weight $w_{f,p,i,j}$, where i and j are the indices of the block, $\mathfrak{r}_{p,i} \triangleq \min(8i + 7, W_p - 1)$ is the right edge of the block, and $\mathfrak{b}_{p,j} \triangleq \min(8j + 7, H_p - 1)$ is the bottom edge of the block. The exact definition of these weights will incorporate many components designed to improve the visual quality of the video compared to a naive optimization for mean-squared error. We will elaborate on the components in turn below.

The choice of 8×8 blocks is a pragmatic one. They should be small enough to allow a meaningful level of control, but large enough to allow efficient implementation. We find that in the presence of quantization noise, we may need 8×8 pixels to reliably estimate local texture or edge directionality or activity in a region. Blocks of this size also provide a 64x reduction in memory bandwidth compared to using per-pixel weights and allow for relatively efficient SIMD.

Finally, we take λ as the master parameter that drives the trade off between quality and bitrate. Importantly, λ is a single, fixed value. Many other works take λ as a free parameter and make adjustments to it in order to model various effects [AN11, ZYLS10]. Taken in isolation, scaling λ is the same as scaling distortion, with

$$\min wD + \lambda R \tag{3}$$

and

$$\min D + w'\lambda R \tag{4}$$

yielding the same optimum if $w' = \frac{1}{w}$. However, as soon as the distortion for two *different* terms is scaled in different ways, this equivalence breaks down.

$$\min w_1 D_1 + w_2 D_2 + \lambda(R_1 + R_2) \tag{5}$$

does not in general yield the same optimum as

$$\min D_1 + D_2 + w_1'\lambda R_1 + w_2'\lambda R_2 \tag{6}$$

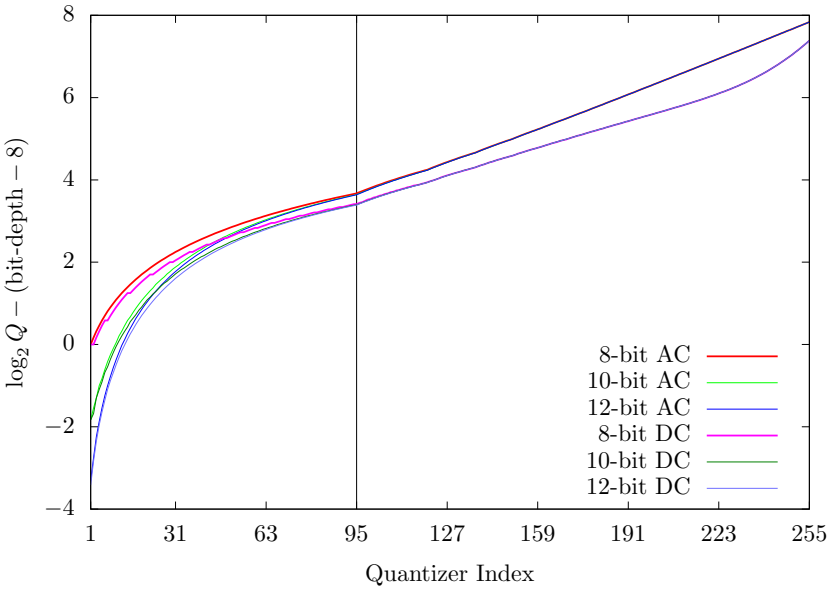

Figure 1: Quantizer vs. quantizer index, logarithmtic scale

for any choice of w_1' or w_2'. There is no way to make these equivalent unless $w_1 = w_2$. If the bits that contribute to R_1 only impact D_1 and the bits that contribute R_2 only impact D_2, then there is no dependency between the two sub-problems, and equivalence is restored. But given the extensive use of prediction, context modeling, and other factors, such sub-problems are rarely truly independent.

So which is the proper approach? We argue that there are many perceptual reasons that one would scale distortion: differences in some pixels are more visible to a human observer than differences in others. However, bits are always bits, and there does not seem to be a reasonable justification for saying that some bits cost more than other bits.

2.1 Quantizer Selection

In AV1, quantizers are table-driven. A *quantizer index* can have one of 256 different values, with 0 representing no quantization (lossless coding), and the rest representing a quantizer chosen from a look-up table. There are separate tables for DC and AC coefficients, with all AC coefficients using the same quantizer by default. These are also separate for 10- and 12-bit video, i.e., these are not simply scaled versions of the table for 8-bit video (or vice versa). VP9 used the same tables. AV1 adopted them without modification.

For 8-bit video, the AC quantizers for quantizer indices from 1 to 95 form a

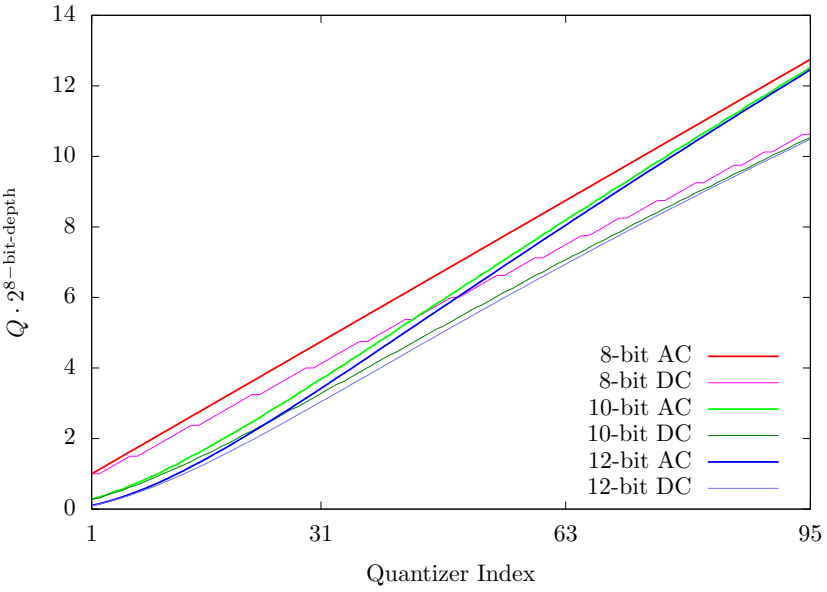

Figure 2: Quantizer vs. quantizer index, linear scale

linear ramp from 1.0 to 12.75[1], as seen in Figure 2. The primary reason for this appears to be that they are stored in the table in Q3 resolution (i.e., in fixed-point with 3 fractional bits), and over this region the difference in consecutive values is $\frac{1}{8}$ (i.e., the minimum representable step size). Beyond quantizer index 95, the quantizers begin incrementing by $\frac{1}{4}$, and then later by $\frac{3}{8}$, etc., such that they grow roughly exponentially, doubling every 36 to 42 steps (i.e., slightly more than 4 times). Figure 1 shows that this region forms an approximately straight line when plotted on a logarithmic scale. These steps look as if they were manually constructed, and not as if they were simply the result of rounding an analytic function.

The DC quantizers are universally smaller than the corresponding AC quantizer (except at 1.0), and do not follow the same consistent patterns. The values in the exponential region instead appear to be fit to data in some fashion, possibly in an attempt to compensate for the tendency of DC coefficients to use the lion's share of the bits at low bitrates, but the reasoning is not documented anywhere to the author's knowledge. The ratio between the quantizers varies, with the DC quantizer on average being 0.767 times the AC quantizer. Because of the limited precision of the table, this means that there are a number of entries where the step from the previous entry is 0 (i.e., the previous entry is repeated).

For 10- and 12-bit, no longer restricted by the resolution of the table, the AC quantizers start off from a value slightly above 1.0 and grow quadratically

[1]For the purposes of discussion, all raw quantizers in the text are scaled assuming transforms with a unit norm. AV1's actual transforms have varying scales depending on the block size.

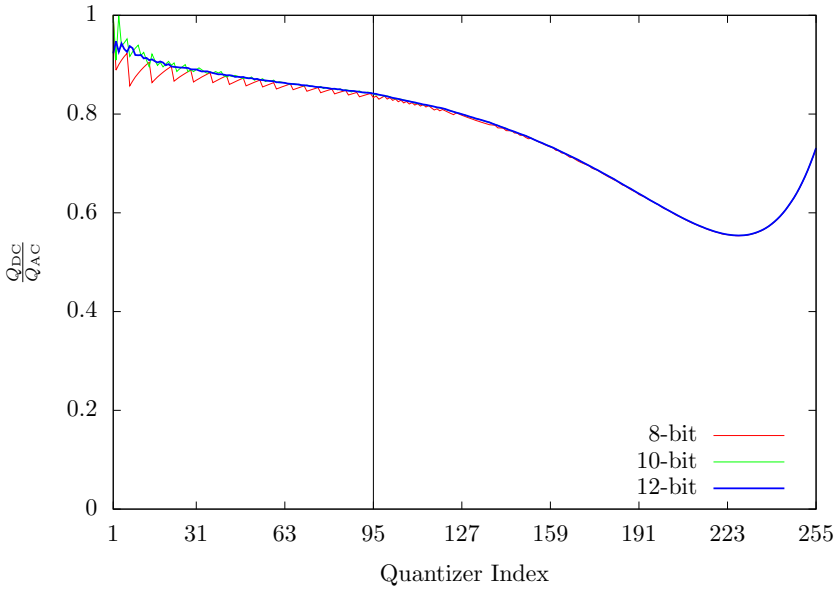

Figure 3: Ratio between DC and AC quantizers

for the first seventeen or so entries before flattening out into a roughly linear region through index 95. The slope does not correspond to that of the 8-bit values, which is likely to compensate for the fact that they start from a different place but end up in very similar places. After that, the exponential section follows the 8-bit table very closely, with values typically between $\frac{13}{8}$ (at the start) to $\frac{1}{8}$ (at the end) smaller than the corresponding 8-bit table scaled up by 16. The discrepancies are large enough to show that the 8-bit values are not simply rounded version of the 12-bit values, but close enough that they clearly are related. They have the same weird discretization of the step sizes at the start. That is, the difference between entries 95 and 96 is roughly double that of entries 94 and 95, despite having ample precision to make this unnecessary. The DC quantizer values appear to be selected to maintain the same ratio with the AC quantizer values at all bit depths, with some notable differences at small quantizers that can be explained by discretization. The effect of the limited precision for the 8-bit values are particularly pronounced, as illustrated in Figure 3.

2.1.1 Flat Quantization

Consider first the simple case where the distortion is not weighted. Under the assumption that residual coefficients are Laplace distributed the optimal choice

of λ for a given quantizer, Q, is

$$\lambda = \frac{\log 2}{6} Q^2 \ . \tag{7}$$

Since the DC and AC quantizers for a given quantizer index differ, it is not obvious how to select a single value of λ suitable for rate-distortion optimization given a quantizer index.

We will return to this problem momentarily, but first consider the opposite problem. Although it is not optimal visually, it is important to be able to configure an encoder to use a flat quantizer (i.e., to quantize all coefficients by the same value), if only for benchmark and testing purposes. We will extend things to non-flat quantization in Section 2.5. A single frame can select different quantization indices for DC and AC in each of the color planes. If we start instead from λ, we can derive the optimal choice of quantizer as

$$Q_{\text{opt}} \triangleq \sqrt{\frac{6}{\log 2} \lambda} \ . \tag{8}$$

It is not in general possible to get a perfectly flat quantizer, as not every value is available in the tables. We can however search the tables for the value closest to Q_{opt} for both DC and AC coefficients.

In order to define 'closest', we use the distance in the log domain. In the limit at high rates, a linear step of quantizer in the log domain leads to a linear change in bitrate, with an increase rate of one bit per pixel each time the quantizer is halved. Therefore the closest quantizer in the log domain should yield a bitrate closest to the optimal rate for λ. Equivalently, if we know Q_{opt} lies between table entries Q_{qi} and Q_{qi+1}, we can compute

$$\log Q_{\text{thresh}} \triangleq \frac{1}{2} \left(\log Q_{qi} + \log Q_{qi+1} \right) \ , \tag{9}$$

$$2 \log Q_{\text{thresh}} = \log Q_{qi} + \log Q_{qi+1}, \tag{10}$$

$$Q^2_{\text{thresh}} = Q_{qi} Q_{qi+1} \tag{11}$$

and choose qi if $Q^2_{\text{opt}} < Q^2_{\text{thresh}}$ and $qi + 1$ otherwise. This allows the search to be implemented in an exact manner with simple arithmetic.

Specifying λ and specifying a quantizer are equivalent, as eqs. 7 and 8 show. However, it may be more intuitive for users to specify the target encoder quality with an abstract quantizer index whose behavior is closer to linear in bitrate. Since an AV1 quantizer index maps to two different quantizers, which do we use to derive λ? The tables clearly seem designed around the AC quantizers, so it would be natural to select them. However, because the DC quantizers are generally smaller, they cannot extend to the full range of values achievable by the AC quantizers. For very large quantizer indices, the deviation from flat quantization becomes quite pronounced, with all quantizer indices from 239 on having an AC quantizer larger than the largest possible DC quantizer. In addition, deriving λ from the AC quantizer means that all of the error due to

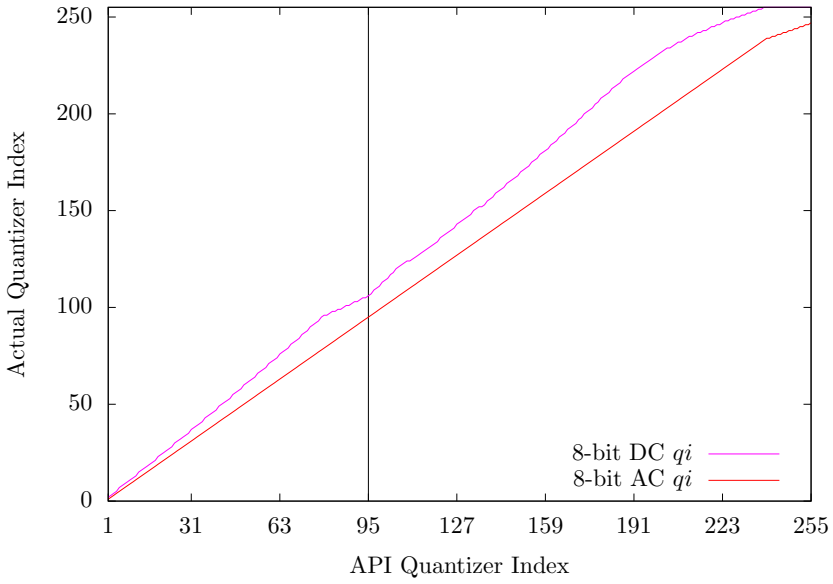

Figure 4: Actual quantization indices used for a given API quantization index in `rav1e` for 8-bit video. Beyond an API quantization index of 80, we are no longer always able to find a perfect match for the AC quantizer in the DC table. At an API quantization index of 239, the DC quantizer index saturates at 255. A larger λ and slightly increased AC quantizer will continue to be able to lower the bitrate, but not as effectively as a larger DC quantizer. The largest AC quantizer index in actual use with our approach is 247.

the discretization of the tables shows up in the DC quantizer. This can lead to suboptimal results at low rates, where often the DC coefficient is the only non-zero coefficient in a block after quantization, but RDO operates with a λ tuned for a (different) AC quantizer. Conversely, for 8-bit video, the fact that multiple quantizer indices map to the same actual DC quantizer makes the DC quantizers less than ideal, especially if the encoder wishes to use multiple quantizers derived as offsets from the base quantizer index.

As a compromise, we first take the AC quantizer corresponding to a given quantizer index, and find the DC quantizer that is closest to it in the table. We then compute Q_{opt} as the geometric average of these quantizers, and use that to derive λ, as Figure 5 illustrates. Finally, we search both the AC and DC quantizer tables for the quantizers closest to Q_{opt}, since both may differ from the originally selected entries, as Figure 4 shows. This distributes the deviation from a perfectly flat quantizer due to the limited table entries available to both the DC and AC coefficients. This is more easily seen in Figure 6.

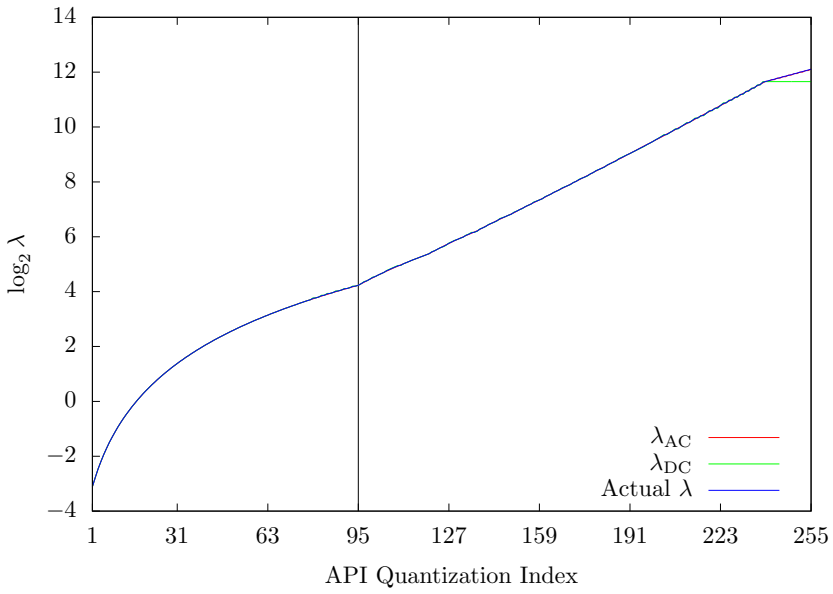

Figure 5: $\log_2 \lambda$ vs. API quantization index in `rav1e` for 8-bit video. Ideally this should be linear, but that would give up useful resolution in the tables for quantizer indices less than 95. The section for quantizer indices greater than 239 is still linear, just with a reduced slope.

2.2 Averaging Quantizers

Once we start adding weights to our distortion measurements, we will also want to use different quantizers for different parts of a frame. In order to support something other than completely uniform quantization of all transform coefficients for all color planes, we need some way to tie our single λ parameter to a collection of different quantizers. This is necessary because it is often not possible to tie particular bits in the coded sequence to particular quantized transform coefficients. For example, the bits spent signaling a a motion vector affect all of the color planes. The bits spent signaling a partition split decision may affect multiple blocks, which in turn might have different weights, and so on.

We approach the problem by considering the entropy of the quantized transform coefficients. Suppose that the transform coefficients follow a zero-mean Laplacian distribution, a choice which balances model simplicity and fidelity [RG83, BCC+92],

$$p(x) = \frac{1}{2b} e^{-\frac{|x|}{b}} \tag{12}$$

$$b \triangleq \sqrt{\frac{\sigma^2}{2}} \ , \tag{13}$$

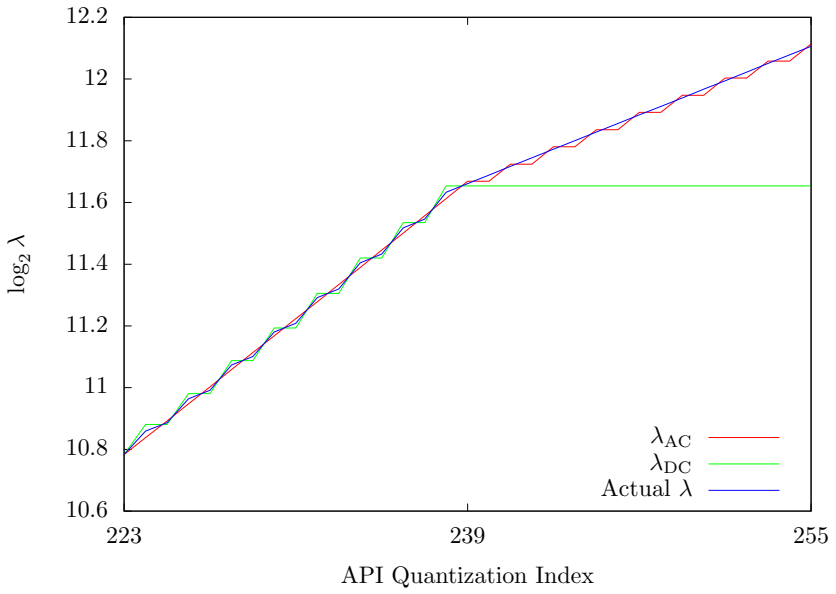

Figure 6: Close-up of Figure 5 for large quantizer indices.

where σ^2 is the variance. Let P_n be the probability that a given coefficient quantizes to the (signed) magnitude k with a given quantizer Q:

$$P_k = \begin{cases} \int_{-(1-\gamma)Q}^{(1-\gamma)Q} p(x)dx, & k = 0 \\ \int_{(k-\gamma)Q}^{(k+1-\gamma)Q} p(x)dx, & k \neq 0 \ . \end{cases} \tag{14}$$

where γ is a rounding bias between 0 and 1. Expanding the integrals yields

$$P_k = \begin{cases} 1 - e^{-(1-\gamma)\frac{Q}{b}}, & k = 0 \\ \frac{1}{2} e^{-(|k|-\gamma+1)\frac{Q}{b}} \left(e^{\frac{Q}{b}} - 1 \right), & k \neq 0 \ . \end{cases} \tag{15}$$

Therefore the entropy of the resulting quantized coefficients is

$$H(Q,b) = -\sum_{k=-\infty}^{\infty} P_k \log_2 P_k \tag{16}$$

$$\begin{aligned} &= -\left(1 - e^{-(1-\gamma)\frac{Q}{b}}\right) \log_2 \left(1 - e^{-(1-\gamma)\frac{Q}{2}}\right) \\ &- 2e^{-(1-\gamma)\frac{Q}{b}} \left[\log_2 \left(1 - e^{-\frac{Q}{b}}\right) + \frac{Q}{b \ln 2} \left(\frac{1}{2} + \frac{1}{1 - e^{-\frac{Q}{b}}} \right) - 1 \right] \ . \end{aligned} \tag{17}$$

This expression is very complicated, but the important observation is that it is characterized entirely by $\frac{Q}{b}$, or equivalently, by $\frac{\sigma^2}{Q^2}$. At low rates, this entropy

is approximately linear in $\frac{\sigma^2}{Q^2}$, and at high rates it is approximately logarithmic in $\frac{\sigma^2}{Q^2}$ [RCN96]. In both cases, as we pool together groups of quantized coefficients with different variances and quantizers, the average value of $\frac{\sigma^2}{Q^2}$ is an important quantity that should be preserved.

Suppose we have multiple groups of transform coefficients, of size N_i and with variances σ_i^2, that will be quantized with different quantizers Q_i. We can consider them to be drawn from a single Laplacian distribution with a single pooled variance

$$\sigma^2_{\text{pooled}} = \frac{\sum_i N_i \sigma_i^2}{\sum_i N_i} \ . \tag{18}$$

This is justified since the theoretical explanation for the Laplacian shape of the transform coefficient distribution in the first place is as the result of pooling coefficients with varying variances [LG00]. Now, we can represent the quantizers used by each group by a single, average quantizer $\bar{Q}$, such that

$$\frac{\sigma^2_{\text{pooled}}}{\bar{Q}^2} = \frac{\sum_i N_i \frac{\sigma_i^2}{Q_i^2}}{\sum_i N_i} \ , \tag{19}$$

which yields

$$\bar{Q} = \sqrt{\frac{\sum_i N_i \sigma_i^2}{\sum_i \frac{N_i \sigma_i^2}{Q_i^2}}} \ . \tag{20}$$

In other words, $\bar{Q}^2$ is just the harmonic mean of the individual Q_i^2 values, weighted by the variance of each group, σ_i^2. We can use this "average quantizer" to select a value of λ via eq. 7. Or, if we have a way of deriving the Q_i values from $\bar{Q}$, then we can select an appropriate $\bar{Q}$ from λ via eq. 8. We give an example of the latter in the next section.

2.2.1 Averaging Weights

It is also informative to turn things around to look at them from the perspective of the weights we are applying to the distortion. Consider a set of N non-negative weights w_i, which could represent the block weights in eq. 2, or per-coefficient weights, or any other form of weighting. Being able to average these weights together is also useful. For example, one might apply a transform that covers several blocks with different weights on their pixel-domain distortions. How should the distortion of the transform coefficients be weighted? It is possible to apply an inverse transform and measure distortion in the pixel domain for each possible rounding direction of each transform coefficient, but this is quite expensive, and the choices are not independent. Similarly, one might wish to know the average weight over an entire frame, in order to pick a base quantizer to work from (even if individual blocks will use varying quantizers).

Assume that each weight w_i corresponds to a quantizer Q_i, which can be chosen independently for the portion of the R-D cost corresponding to w_i,

$$C_i = w_i D + \lambda R \ . \tag{21}$$

Then by eq. 8, the optimal choice for Q_i is

$$Q_i = \sqrt{\frac{6}{\log 2}\frac{\lambda}{w_i}} \ . \tag{22}$$

Define a base quantizer

$$Q_{\text{base}} \triangleq \sqrt{\frac{6}{\log 2}\lambda} \tag{23}$$

as the optimal choice for the cost with an unweighted λ, so that

$$Q_i = \frac{Q_{\text{base}}}{\sqrt{w_i}} \ . \tag{24}$$

The exact value of Q_{base} (and thus λ) is mostly immaterial here, we just use it to simplify the expressions that follow. The important thing is to capture the effect weighting the distortion has on the choice of quantizer, as expressed in eq. 24.

Now suppose we want to define an "average weight", $\bar{w}$, analgous to our "average quantizer", $\bar{Q}$, such that

$$\bar{Q} = \frac{Q_{\text{base}}}{\sqrt{\bar{w}}} \ . \tag{25}$$

Consider eq. 20, and assume that the size of each group, N_i, and the variances of the prediction residual, σ_i, are equal for all i, so they can be dropped for simplicity. Then plugging in eqs. 24 and 25 and solving for $\bar{w}$ yields

$$\bar{Q} = \sqrt{\frac{N}{\sum_i \frac{1}{Q_i^2}}} \tag{26}$$

$$\frac{Q_{\text{base}}}{\sqrt{\bar{w}}} = \sqrt{\frac{N}{\sum_i \frac{w_i}{Q_{\text{base}}^2}}} \tag{27}$$

$$\frac{Q_{\text{base}}^2}{\bar{w}} = \frac{N}{\sum_i \frac{w_i}{Q_{\text{base}}^2}} \tag{28}$$

$$\frac{Q_{\text{base}}^2}{\bar{w}} = \frac{N}{\frac{1}{Q_{\text{base}}^2}\sum_i w_i} \tag{29}$$

$$\frac{1}{\bar{w}} = \frac{N}{\sum_i w_i} \tag{30}$$

$$\bar{w} = \frac{\sum_i w_i}{N} \ . \tag{31}$$

That is, the "average weight" that is consistent with our theory for averaging quantizers really is just the normal arithmetic mean.

2.3 Color

The perceptual impact of the chroma planes is not equal to that of the luma planes, and the corresponding weight of each plane must be tuned. The tuning of the balance between chroma and luma in rav1e has been described previously [BBM+19], but is reviewed here for completeness. At the threshold of perception, the eye is more sensitive to changes in luma than changes in chroma, and we can use a coarser quantizer for chroma. At low bitrates (far from the threshold of perception), luma loses most of its high-frequency components, making chroma differences more noticeable. At these rates chroma requires a finer quantizer than luma.

We use the CIEDE2000 metric [CIE01, SWD05] to evaluate the impact of coding artifacts in all three color planes on color quality. This metric is based on the CIEL*a*b* color space [CIE19] with corrections to improve perceptual uniformity. However, visual inspection shows that optimizing the luma-chroma balance solely for CIEDE2000 causes a noticable loss in luma details, which is confirmed by a large drop in luma PSNR. Therefore, we tune rav1e for the (equally-weighted) sum of CIEDE2000 and luma PSNR, choosing parameters such that the first derivatives of BD-rate [Bjø01] with each metric have equal (and opposite) magnitudes. This causes only marginal losses in CIEDE2000 at the extremums, but significantly reduces the losses in luma PSNR. These tunings agree very closely with earlier tuning performed for Daala [daa], a completely different video codec with very different perceptual properties, suggesting they are relatively robust to other aspects of a codec or encoder implementation.

The color plane weights are fixed at high bitrates and scaled linearly with λ. We have not experimentally verified that a model that is linear in λ accurately describes the change in weights required to jointly minimize BD-rate for CIEDE2000 and luma PSNR across the full bitrate range.

In order to determine the quantizers for an overall λ (or, equivalently, the overall λ for a given set of quantizers, c.f. Section 2.2), we need to know the the average variance of the prediction residual in each color plane. We encode four 720p sequences (`ducks_take_off`, `in_to_tree`, `old_town_cross`, and `park_joy`), chosen because we have 4:2:0, 4:2:2, and 4:4:4 versions of each of them. We encode at all API quantization indices $1 \ldots 239$ with the default speed level. We use low-latency mode (no bi-predicted frames), we disable temporal RDO, and optimize for PSNR, to ensure constant quantizers. During each encode, we collect the sum and sum of squares of the prediction residual in each partition, and record only the final encoding parameters selected by RDO (i.e., the prediction mode that is actually used to encode the partition). These sums are used to compute the variance for the DC and AC coefficients of each color plane.

Some preliminary graphs follow. Figure 7 shows the absolute variances of each component for partitions encoded in inter modes, plotted against the log of the quantizer. Figure 8 shows the relative variances of each component for partitions encoded in inter modes, plotted against the log of the luma quantizer (the corresponding chroma quantizers vary, as described above). This data needs to be reprocessed, as the current mechanism for averaging the variance per

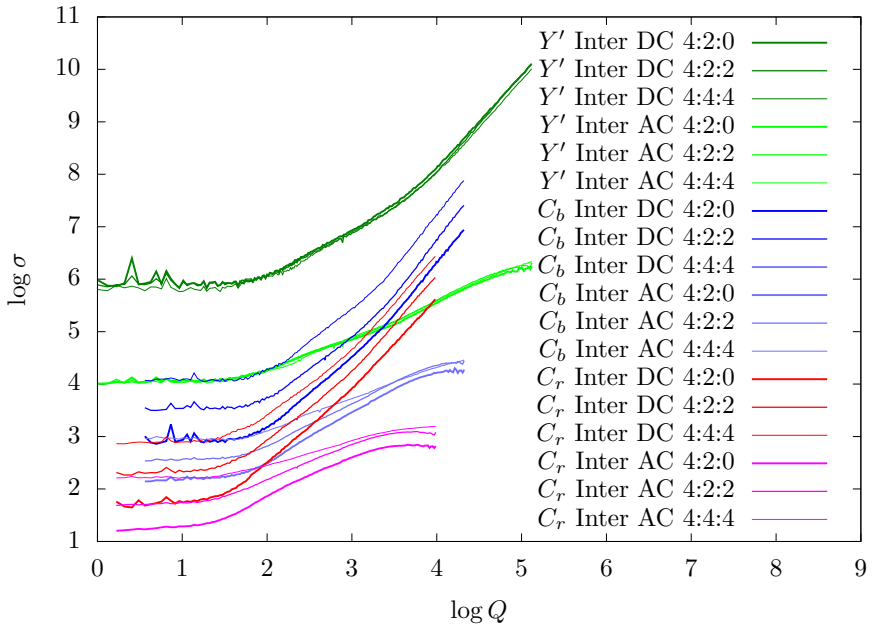

Figure 7: Color plane variances (Inter modes).

quantizer across different encodes is subtly broken. Figure 9 shows the absolute variances of each component for 4:2:0 partitions encoded in intra modes, plotted against the quantizer. This data is clearly broken. At least two issues with the current data collection approach are suspected of contributing to this: the same averaging issue described above, and the fact that the statistics of intra blocks in intra-only frames are very different from intra blocks in inter frames, due to the fact that in the latter they must compete against other inter modes in order to be selected.

2.4 Adaptive Quantization

2.4.1 Activity Masking

2.4.2 Temporal RDO

2.5 Quantization Matrices

2.6 Rate Control

References

[AN11] Cheolhong An and Truong Q. Nguyen. Adaptive Lagrange multiplier selection using classification-maximization and its application to

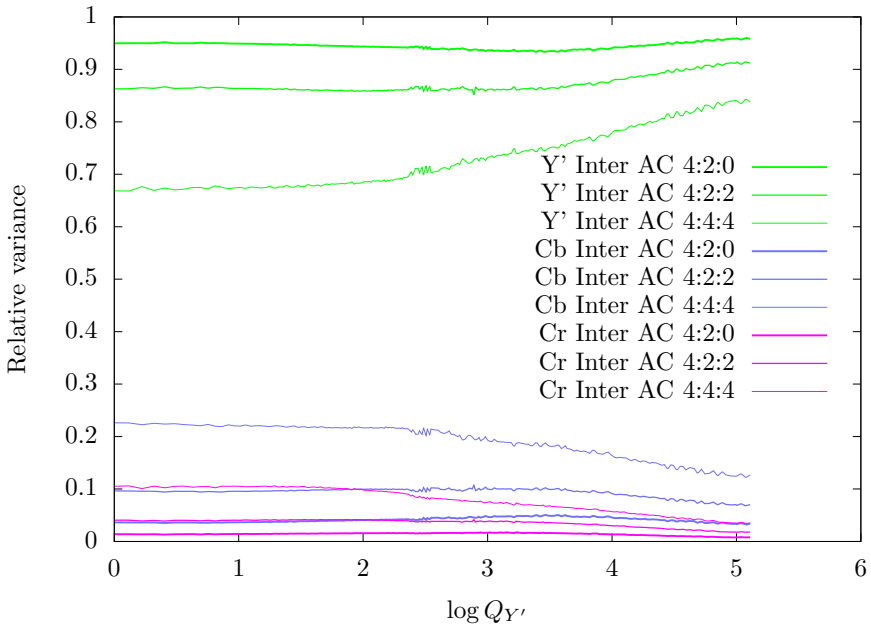

Figure 8: Relative plane variances (Inter modes).

chroma QP offset decision. *IEEE Transactions on Circuits and Systems for Video Technology*, 21(6):783–791, June 2011.

[BBM+19] Luca Barbato, David M. Barr, Ivan Molodetskikh, Christopher Montomery, S. P. Shreevari, Raphaël A. Zumer, and Nathan E. Egge. Rust AV1 encoder (rav1e) project. In *Proceedings of the SPIE: Applications of Digital Image Processing XLII (ADIP'19)*, volume 11137, San Diego, September 2019.

[BCC+92] Fabio L. Bellifemine, A. Capellino, Antonio Chimienti, Romualdo Picco, and Remo Ponti. Statiscial analysis of the 2D-DCT coefficients of the differential signal for images. *Signal Processing: Image Communications*, 4(6):477–488, November 1992.

[Bjø01] Gisle Bjøntegaard. Calculation of average PSNR differences between RD curves. Technical Report VCEG-M33, ITU-T SG16/Q6, Austin, TX, April 2001.

[CIE01] Improvement to industrial color-difference evaluation. Technical Report 142-2001, Commission International de l'Éclairage, Vienna, 2001.

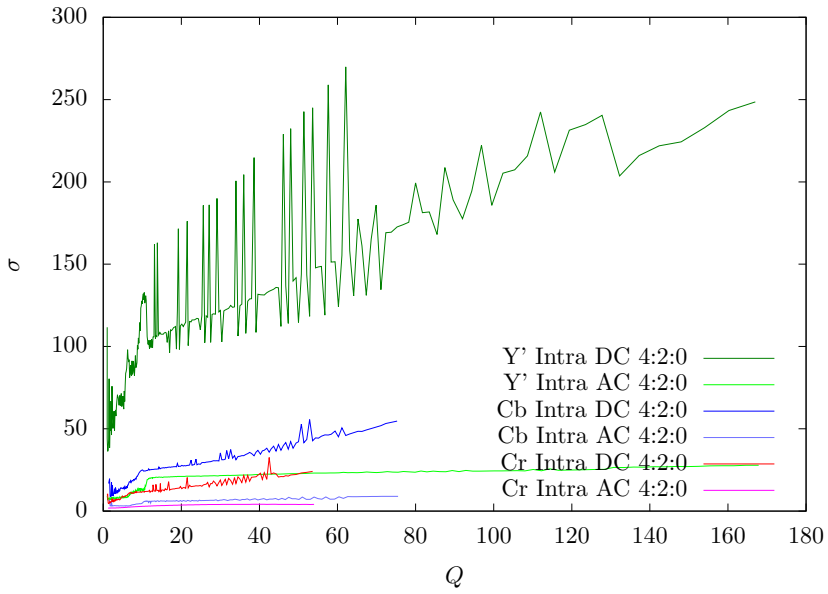

Figure 9: Color plane variances (Intra 4:2:0).

[CIE19] Colorimetry — part 4: CIE L*a*b* color space. Technical Report ISO/CIE 11664-4:2019(E), Commission International de l'Éclairage, 2019.

[daa] Daala website. https://xiph.org/daala/.

[LG00] Edmund Y. Lam and Joseph W. Goodman. A mathematical analysis of the DCT coefficient distributions for images. *IEEE Transactions on Image Processing*, 9(10):1661–1666, October 2000.

[RCN96] Jordi Ribas-Corbera and David L. Neuhoff. On the optimal motion vector accuracy for block-based motion-compensated video coders. In *Digital Video Compression: Algorithms and Technologies*, volume 2268 of *Proceedings of the SPIE*, San Jose, CA, March 1996.

[RG83] Randall C. Reiningek and Jerry D. Gibson. Distributions of the two-dimensional DCT coefficients for images. *IEEE Transactions on Communications*, 31(6):835–839, June 1983.

[SWD05] Gaurav Sharma, Wencheng Wu, and Edul N. Dalal. The CIEDE2000 color-difference formula: Implementation notes, supplementary test data, and mathematical observations. *Color Research and Application*, 30(1):21–30, February 2005.

[ZYLS10] Jun Zhang, Xiaoquan Yi, Nam Ling, and Weijia Shang. Context adaptive Lagrange multiplier (CALM) for rate-distortion optimal

motion estimation in video coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 20(6):820–828, June 2010.