.oOo.

The Opus Codec

To be presented at the 135th AES Convention
2013 October 17–20 New York, USA

This paper was accepted for publication at the 135th AES Convention. This version of the paper is from the authors and not from the AES.

Voice Coding with Opus

Koen Vos, Karsten Vandborg Sørensen[1], Søren Skak Jensen[2], and Jean-Marc Valin[3]

[1] *Microsoft, Applications and Services Group, Audio DSP Team, Stockholm, Sweden*

[2] *GN Netcom A/S, Ballerup, Denmark*

[3] *Mozilla Corporation, Mountain View, CA, USA*

Correspondence should be addressed to Koen Vos (`koenvos74@gmail.com`)

ABSTRACT

In this paper, we describe the voice mode of the Opus speech and audio codec. As only the decoder is standardized, the details in this paper will help anyone who wants to modify the encoder or gain a better understanding of the codec. We go through the main components that constitute the voice part of the codec, provide an overview, give insights, and discuss the design decisions made during the development. Tests have shown that Opus quality is comparable to or better than several state-of-the-art voice codecs, while covering a much broader application area than competing codecs.

1. INTRODUCTION

The Opus speech and audio codec [1] was standardized by the IETF as RFC6716 in 2012 [2]. A companion paper [3], gives a high-level overview of the codec and explains its music mode. In this paper we discuss the voice part of Opus, and when we refer to Opus we refer to Opus in the voice mode only, unless explicitly specified otherwise.

Opus is a highly flexible codec, and in the following we outline the modes of operation. We only list what is supported in voice mode.

- Supported sample rates are shown in Table 1.
- Target bitrates down to 6 kbps are supported. Recommended bitrates for different sample rates are shown in Table 2.
- The frame duration can be 10 and 20 ms, and for NB, MB, and WB, there is also support for 40 and 60 ms, where 40 and 60 ms are concatenations of 20 ms frames with some of the coding of the concatenated frames being conditional.
- Complexity mode can be set from 0-10 with 10 being the most complex mode.

Opus has several control options specifically for voice applications:

Sample Frequency	Name	Acronym
48 kHz	Fullband	FB
24 kHz	Super-wideband	SWB
16 kHz	Wideband	WB
12 kHz	Mediumband	MB
8 kHz	Narrowband	NB

Table 1: Supported sample frequencies.

Input	Recommended Bitrate Range	
Type	Mono	Stereo
FB	28-40 kbps	48-72 kbps
SWB	20-28 kbps	36-48 kbps
WB	16-20 kbps	28-36 kbps
MB	12-16 kbps	20-28 kbps
NB	8-12 kbps	14-20 kbps

Table 2: Recommended bitrate ranges.

- Discontinuous Transmission (DTX). This reduces the packet rate when the input signal is classified as silent, letting the decoder's Packet-Loss Concealment (PLC) fill in comfort noise during the non-transmitted frames.
- Forward Error Correction (FEC). To aid packet-loss robustness, this adds a coarser description of a packet to the next packet. The decoder can use the coarser description if the earlier packet with the main description was lost, provided the jitter buffer latency is sufficient.
- Variable inter-frame dependency. This adjusts the dependency of the Long-Term Predictor (LTP) on previous packets by dynamically down scaling the LTP state at frame boundaries. More down scaling gives faster convergence to the ideal output after a lost packet, at the cost of lower coding efficiency.

The remainder of the paper is organized as follows: In Section 2 we start by introducing the coding models. Then, in Section 3, we go though the main functions in the encoder, and in Section 4 we briefly go through the decoder. We then discuss listening results in Section 5 and finally we provide conclusions in Section 6.

2. CODING MODELS

The Opus standard defines models based on the Modified Discrete Cosine Transform (MDCT) and on Linear-Predictive Coding (LPC). For voice signals, the LPC model is used for the lower part of the spectrum, with the MDCT coding taking over above 8 kHz. The LPC based model is based on the SILK codec, see [4]. Only frequency bands between 8 and (up to) 20 kHz[1] are coded with MDCT. For details on the MDCT-based model, we refer to [3].

As evident from Table 3 there are no frequency ranges for which both models are in use.

Sample	Frequency Range	
Frequency	LPC	MDCT
48 kHz	0-8 kHz	8-20 kHz[1]
24 kHz	0-8 kHz	8-12 kHz
16 kHz	0-8 kHz	-
12 kHz	0-6 kHz	-
8 kHz	0-4 kHz	-

Table 3: Model uses at different sample frequencies, for voice signals.

The advantage of using a hybrid of these two models is that for speech, linear prediction techniques, such as Code-Excited Linear Prediction (CELP), code low frequencies more efficiently than transform (e.g., MDCT) domain techniques, while for high speech frequencies this advantage diminishes and transform coding has better numerical and complexity characteristics. A codec that combines the two models can achieve better quality at a wider range of sample frequencies than by using either one alone.

3. ENCODER

The Opus encoder operates on frames of either 10 or 20 ms, which are divided into 5 ms subframes. The following paragraphs describe the main components of the encoder. We refer to Figure 1 for an overview of how the individual functions interact.

3.1. VAD

The Voice Activity Detector (VAD) generates a measure of speech activity by combining the signal-to-noise ratios (SNRs) from 4 separate frequency bands.

[1] Opus never codes audio above 20 kHz, as that is the upper limit of human hearing.

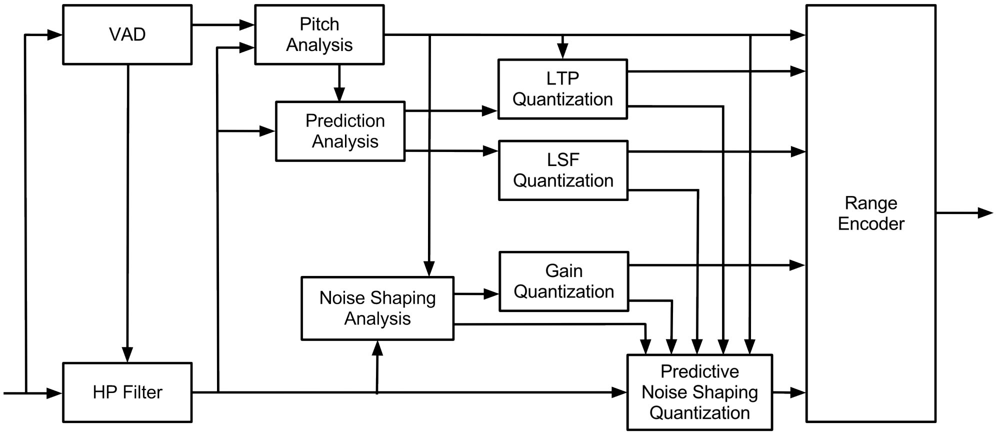

Fig. 1: Encoder block diagram.

In each band the background noise level is estimated by smoothing the inverse energy over time frames. Multiplying this smoothed inverse energy with the subband energy gives the SNR.

3.2. HP Filter

A high-pass (HP) filter with a variable cutoff frequency between 60 and 100 Hz removes low-frequency background and breathing noise. The cutoff frequency depends on the SNR in the lowest frequency band of the VAD, and on the smoothed pitch frequencies found in the pitch analysis, so that high pitched voices will have a higher cutoff frequency.

3.3. Pitch Analysis

As shown in Figure 2, the pitch analysis begins by pre-whitening the input signal, with a filter of order between 6 and 16 depending the the complexity mode. The whitening makes the pitch analysis equally sensitive to all parts of the audio spectrum, thus reducing the influence of a strong individual harmonic. It also improves the accuracy of the correlation measure used later to classify the signal as voiced or unvoiced.

The whitened signal is then downsampled in two steps to 8 and 4 kHz, to reduce the complexity of computing correlations. A first analysis step finds peaks in the autocorrelation of the most downsampled signal to obtain a small number of coarse pitch lag candidates. These are input to a finer analysis step running at 8 kHz, searching only around the preliminary estimates. After applying a small bias towards shorter lags to avoid pitch doubling, a single candidate pitch lag with highest correlation is found.

The candidate's correlation value is compared to a threshold that depends on a weighted combination of:

- Signal type of the prevous frame.
- Speech activity level.
- The slope of the SNR found in the VAD with respect to frequency.

If the correlation is below the threshold, the signal is classified as unvoiced and the pitch analysis is aborted without returning a pitch lag estimate.

The final analysis step operates on the input sample frequency (8, 12 or 16 kHz), and searches for integer-sample pitch lags around the previous stage's estimate, limited to a range of 55.6 to 500 Hz . For each lag being evaluated, a set of pitch contours from a codebook is tested. These pitch contours define a deviation from the average pitch lag per 5 ms subframe, thus allowing the pitch to vary within a frame. Between 3 and 34 pitch contour vectors are available, depending on the sampling rate and frame size. The pitch lag and contour index resulting in the highest correlation value are encoded and transmitted to the decoder.

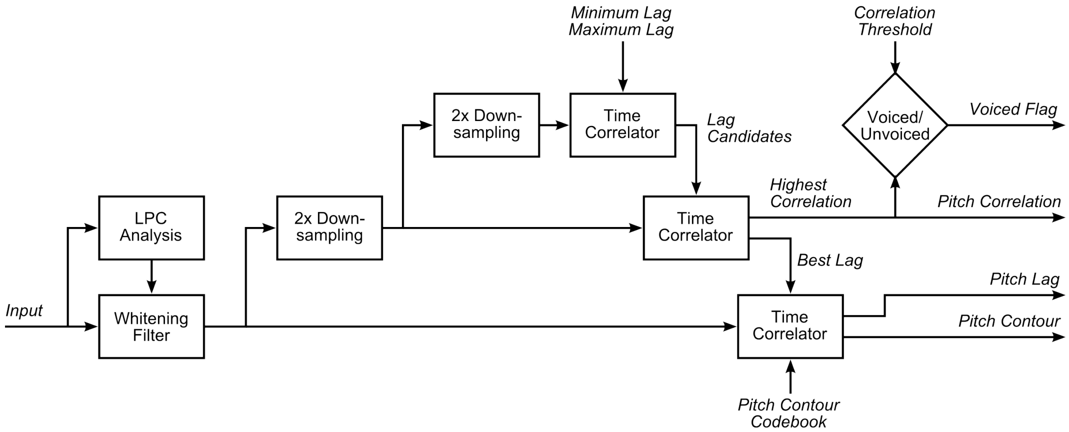

Fig. 2: Block diagram of the pitch analysis.

3.3.1. Correlation Measure

Most correlation-based pitch estimators normalize the correlation with the geometric mean of the energies of the vectors being correlated:

$$C = \frac{\mathbf{x}^T\mathbf{y}}{\sqrt{(\mathbf{x}^T\mathbf{x} \cdot \mathbf{y}^T\mathbf{y})}}, \tag{1}$$

whereas Opus normalizes with the *arithmetic* mean:

$$C_{Opus} = \frac{\mathbf{x}^T\mathbf{y}}{\frac{1}{2}(\mathbf{x}^T\mathbf{x} + \mathbf{y}^T\mathbf{y})}. \tag{2}$$

This correlation measures similarity not just in shape, but also in scale. Two vectors with very different energies will have a lower correlation, similar to frequency-domain pitch estimators.

3.4. Prediction Analysis

As described in Section 3.3, the input signal is prewhitened as part of the pitch analysis. The prewhitened signal is passed to the prediction analysis in addition to the input signal. The signal at this point is classified as being either voiced or unvoiced. We describe these two cases in Section 3.4.1 and 3.4.2.

3.4.1. Voiced Speech

The long-term prediction (LTP) of voiced signals is implemented with a fifth order filter. The LTP coefficients are estimated from the pre-whitened input signal with the covariance method for every 5 ms subframe. The coefficients are quantized and used to filter the input signal (without pre-whitening) to find an LTP residual. This signal is input to the LPC analysis, where Burg's method [5], is used to find short-term prediction coefficients. Burg's method provides higher prediction gain than the autocorrelation method and, unlike the covariance method, it produces stable filter coefficients. The LPC order is $N_{LPC} = 16$ for FB, SWB, and WB, and $N_{LPC} = 10$ for MB and NB. A novel implementation of Burg's method reduces its complexity to near that of the autocorrelation method [6]. Also, the signal in each sub-frame is scaled by the inverse of the quantization step size in that sub-frame before applying Burg's method. This is done to find the coefficients that minimize the number of bits necessary to encode the residual signal of the frame rather than minimizing the energy of the residual signal.

Computing LPC coefficients based on the LTP residual rather than on the input signal approximates a joint optimization of these two sets of coefficients [7]. This increases the prediction gain, thus reducing the bitrate. Moreover, because the LTP prediction is typically most effective at low frequencies, it reduces the dynamic range of the AR spectrum defined by the LPC coefficients. This helps with the numerical properties of the LPC analysis and filtering, and avoids the need for any pre-emphasis filtering found in other codecs.

3.4.2. Unvoiced Speech

For unvoiced signals, the pre-whitened signal is discarded and Burg's method is used directly on the input signal.

The LPC coefficients (for either voiced or unvoiced speech) are converted to Line Spectral Frequencies (LSFs), quantized and used to re-calculate the LPC residual taking into account the LSF quantization effects. Section 3.7 describes the LSF quantization.

3.5. Noise Shaping

Quantization noise shaping is used to exploit the properties of the human auditory system.

A typical state-of-the-art speech encoder determines the excitation signal by minimizing the perceptually-weighted reconstruction error. The decoder then uses a postfilter on the reconstructed signal to suppress spectral regions where the quantization noise is expected to be high relative to the signal. Opus combines these two functions in the encoder's quantizer by applying different weighting filters to the input and reconstructed signals in the noise shaping configuration of Figure 3. Integrating the two operations on the encoder side not only simplifies the decoder, it also lets the encoder use arbitrarily simple or sophisticated perceptual models to simultaneously and independently shape the quantization noise and boost/suppress spectral regions. Such different models can be used without spending bits on side information or changing the bitstream format. As an example of this, Opus uses warped noise shaping filters at higher complexity settings as the frequency-dependent resolution of these filters better matches human hearing [8]. Separating the noise shaping from the linear prediction also lets us select prediction coefficients that minimize the bitrate without regard for perceptual considerations.

A diagram of the Noise Shaping Quantization (NSQ) is shown in Figure 3. Unlike typical noise shaping quantizers where the noise shaping sits directly around the quantizer and feeds back to the input, in Opus the noise shaping compares the input and output speech signals and feeds to the input of the quantizer. This was first proposed in Figure 3 of [9]. More details of the NSQ module are described in Section 3.5.2.

3.5.1. Noise Shaping Analysis

The Noise Shaping Analysis (NSA) function finds gains and filter coefficients used by the NSQ to shape the signal spectrum with the following purposes:

- Spectral shaping of the quantization noise similarly to the speech spectrum to make it less audible.
- Suppressing the spectral valleys in between formant and harmonic peaks to make the signal less noisy and more predictable.

For each subframe, a quantization gain (or step size) is chosen and sent to the decoder. This quantization gain determines the tradeoff between quantization noise and bitrate.

Furthermore, a compensation gain and a spectral tilt are found to match the decoded speech level and tilt to those of the input signal.

The filtering of the input signal is done using the filter

$$H(z) = G \cdot (1 - c_{tilt} \cdot z^{-1}) \cdot \frac{W_{ana}(z)}{W_{syn}(z)}, \tag{3}$$

where G is the compensation gain, and c_{tilt} is the tilt coefficient in a first order tilt adjustment filter. The analysis filter are for voiced speech given by

$$\begin{aligned} W_{ana}(z) &= \left(1 - \sum_{k=1}^{N_{LPC}} a_{ana}(k) \cdot z^{-k}\right) & (4)\\ &\cdot \left(1 - z^{-L} \cdot \sum_{k=-2}^{2} b_{ana}(k) \cdot z^{-k}\right), & (5)\end{aligned}$$

and similarly for the synthesis filter $W_{syn}(z)$. N_{LPC} is the LPC order and L is the pitch lag in samples. For unvoiced speech, the last term (5) is omitted to disable harmonic noise shaping.

The short-term noise shaping coefficients $a_{ana}(k)$ and $a_{syn}(k)$ are calculated from the LPC of the input signal $a(k)$ by applying different amounts of bandwidth expansion, i.e.,

$$\begin{aligned} a_{ana}(k) &= a(k) \cdot g_{ana}^{k}, \text{ and} & (6)\\ a_{syn}(k) &= a(k) \cdot g_{syn}^{k}. & (7)\end{aligned}$$

The bandwidth expansion moves the roots of the LPC polynomial towards the origin, and thereby flattens the spectral envelope described by $a(k)$.

The bandwidth expansion factors are given by

$$\begin{aligned} g_{ana} &= 0.95 - 0.01 \cdot C, \text{ and} & (8)\\ g_{syn} &= 0.95 + 0.01 \cdot C, & (9)\end{aligned}$$

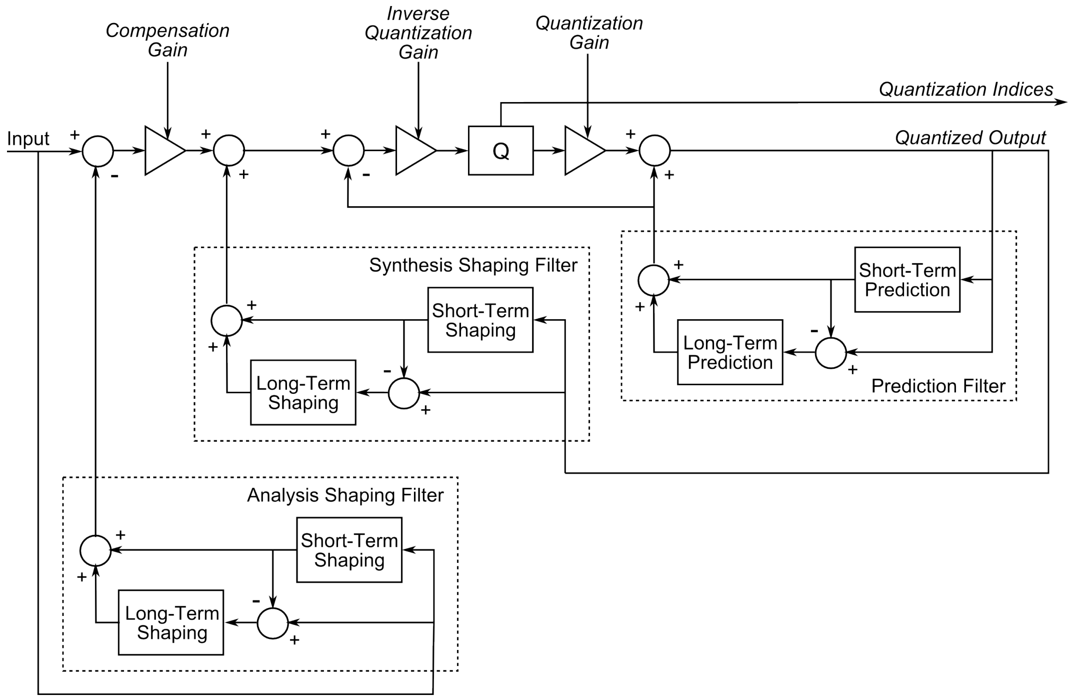

Fig. 3: Predictive Noise Shaping Quantizer.

where $C \in [0, 1]$ is a coding quality control parameter. By applying more bandwidth expansion to the analysis part than the synthesis part, we de-emphasize the spectral valleys.

The harmonic noise shaping applied to voiced frames has three filter taps

$$\begin{aligned} b_{ana} &= F_{ana} \cdot [0.25, 0.5, 0.25], \text{ and} \quad (10) \\ b_{syn} &= F_{syn} \cdot [0.25, 0.5, 0.25], \quad (11) \end{aligned}$$

where the multipliers F_{ana} and $F_{syn} \in [0, 1]$ are calculated from:

- The coding quality control parameter. This makes the decoded signal more harmonic, and thus easier to encode, at low bitrates.
- Pitch correlation. Highly periodic input signal are given more harmonic noise shaping to avoid audible noise between harmoncis.
- The estimated input SNR below 1 kHz. This filters out background noise for a noise input signal by applying more harmonic emphasis.

Similar to the short-term shaping, having $F_{ana} < F_{syn}$ emphasizes pitch harmonics and suppresses the signal in between the harmonics.

The tilt coefficient c_{tilt} is calculated as

$$c_{tilt} = 0.25 + 0.2625 \cdot V, \quad (12)$$

where $V \in [0, 1]$ is a voice activity level which, in this context, is forced to 0 for unvoiced speech.

Finally, the compensation gain G is calculated as the ratio of the prediction gains of the short-term prediction filters a_{ana} and a_{syn}.

An example of short-term noise shaping of a speech spectrum is shown in Figure 4. The weighted input and quantization noise combine to produce an output with spectral envelope similar to the input signal.

3.5.2. Noise Shaping Quantization

The NSQ module quantizes the residual signal and thereby generates the excitation signal.

A simplified block diagram of the NSQ is shown in Figure 5. In this figure, $P(z)$ is the predictor con-

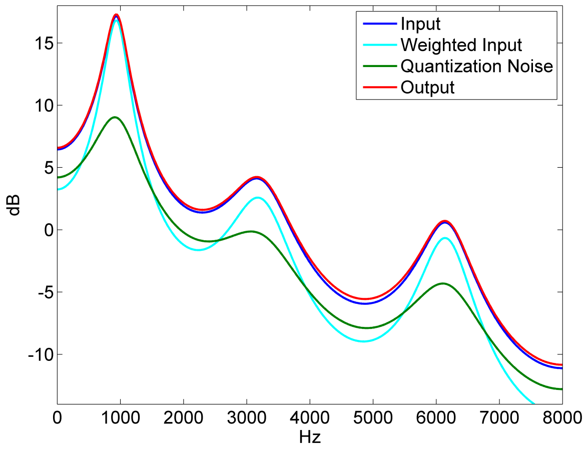

Fig. 4: Example of how the noise shaping operates on a speech spectrum. The frame is classified as unvoiced for illustrative purposes, showing only short-term noise shaping.

taining both the LPC and LTP filters. $F_{ana}(z)$ and $F_{syn}(z)$ are the analysis and synthesis noise shaping filters, and for voiced speech they each consist of both long term and short term filters. The quantized excitation indices are denoted $i(n)$. The LTP coefficients, gains, and noise shaping coefficients are updated for every subframe, whereas the LPC coefficients are updated every frame.

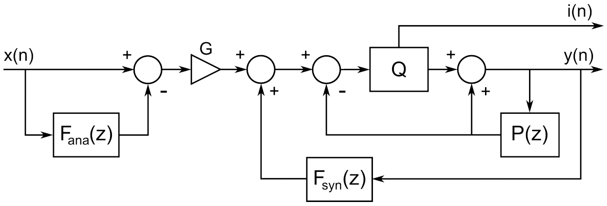

Fig. 5: Noise Shaping Quantization block diagram.

Substituting the quantizer Q with addition of a quantization noise signal $q(n)$, the output of the NSQ is given by:

$$Y(z) = G \cdot \frac{1 - F_{ana}(z)}{1 - F_{syn}(z)} \cdot X(z) + \frac{1}{1 - F_{syn}(z)} \cdot Q(z) \tag{13}$$

The first part of the equation is the input signal shaping part and the second part is the quantization noise shaping part.

3.5.3. Trellis Quantizer

The quantizer Q in the NSQ block diagram is a trellis quantizer, implemented as a uniform scalar quantizer with a variable offset. This offset depends on the output of a pseudorandom generator, implemented with linear congruent recursions on previous quantization decisions within the same frame [12]. Since the quantization error for each residual sample now depends on previous quantization decisions, both because of the trellis nature of the quantizer and through the shaping and prediction filters, improved R-D performance is achieved by implementing a Viterbi delayed decision mechanism [13]. The number of different Viterbi states to track, $N \in [2, 4]$, and the number of samples delay, $D \in [16, 32]$, are functions of the complexity setting. At the lowest complexity levels each sample is simply coded independently.

3.6. Pulse Coding

The integer-valued excitation signal which is the output from the NSQ is entropy coded in blocks of 16 samples. First the signal is split into its absolute values, called pulses, and signs. Then the total sum of pulses per block are coded. Next we repeatedly split each block in two equal parts, each time encoding the allocation of pulses to each half, until subblocks either have length one or contain zero pulses. Finally the signs for non-zero samples are encoded separately. The range coding tables for the splits are optimized for a large training database.

3.7. LSF Quantization

The LSF quantizer consists of a VQ stage with 32 codebook vectors followed by a scalar quantization stage with inter-LSF prediction. All quantization indices are entropy coded, and the entropy coding tables selected for the second stage depend on the quantization index from the first. Consequently, the LSQ quantizer uses variable bitrate, which lowers the average R-D error, and reduce the impact of outliers.

3.7.1. Tree Search

As proposed in [14], the error signals from the N best quantization candidates from the first stage are all used as input for the next stage. After the second stage, the best combined path is chosen. By varying the number N, we get a means for adjusting the trade-off between a low rate-distortion (R-D) error and a high computational complexity. The same principle is used in the NSQ, see Section 3.5.3.

3.7.2. Error Sensitivity

Whereas input vectors to the first stage are unweighted, the residual input to the second stage is scaled by the square roots of the Inverse Harmonic Mean Weights (IHMWs) proposed by Laroia et al. in [10]. The IHMWs are calculated from the coarsely-quantized reconstruction found in the first stage, so that encoder and decoder can use the exact same weights. The application of the weights partially normalizes the error sensitivity for the second stage input vector, and it enables the use of a uniform quantizer with fixed step size to be used without too much loss in quality.

3.7.3. Scalar Quantization

The second stage uses predictive delayed decision scalar quantization. The predictor multiplies the previous quantized residual value by a prediction coefficient that depends on the vector index from the first stage codebook as well as the index for the current scalar in the residual vector. The predicted value is subtracted from the second stage input value before quantization and is added back afterwards. This creates a dependency for the current decision on the previous quantization decision, which again is exploited in a Viterbi-like delayed-decision algorithm to choose the sequence of quantization indices yielding the lowest R-D.

3.7.4. GMM interpretation

The LSF quantizer has similarities with a Gaussian mixture model (GMM) based quantizer [15], where the first stage encodes the mean and the second stage uses the Cholesky decomposition of a tridiagonal approximation of the correlation matrix. What is different is the scaling of the residual vector by the IHMWs, and the fact that the quantized residuals are entropy coded with a entropy table that is trained rather than Gaussian.

3.8. Adaptive Inter-Frame Dependency

The presence of long term prediction, or an Adaptive Codebook, is known to give challenges when packet losses occur. The problem with LTP prediction is due to the impulse response of the filter which can be much longer than the packet itself.

An often used technique is to reduce the LTP coefficients, see e.g. [11], which effectively shortens the impulse response of the LTP filter.

We have solved the problem in a different way; in Opus the LTP filter state is downscaled in the beginning of a packet and the LTP coefficients are kept unchanged. Downscaling the LTP state reduces the LTP prediction gain only in the first pitch period in the packet, and therefore extra bits are only needed for encoding the higher residual energy during that first pitch period. Because of Jensens inequality, its better to fork out the bits upfront and be done with it. The scaling factor is quantized to one of three values and is thus transmitted with very few bits.

Compared to scaling the LTP coefficients, downscaling the LTP state gives a more efficient trade-off between increased bit rate caused by lower LTP prediction gain and encoder/decoder resynchronization speed which is illustrated in Figure 6.

3.9. Entropy Coding

The quantized parameters and the excitation signal are all entropy coded using range coding, see [17].

3.10. Stereo Prediction

In Stereo mode, Opus uses predictive stereo encoding [16] where it first encodes a mid channel as the average of the left and right speech signals. Next it computes the side channel as the difference between left and right, and both mid and side channels are split into low- and high-frequency bands. Each side channel band is then predicted from the corresponding mid band using a scalar predictor. The prediction-residual bands are combined to form the side residual signal S, which is coded independently from the mid channel M. The full approach is illustrated in Figure 7. The decoder goes through these same steps in reverse order.

4. DECODING

The predictive filtering consist of LTP and LPC. As shown in Figure 8, it is implemented in the decoder through the steps of parameter decoding, constructing the excitation, followed by long-term and short-term synthesis filtering. It has been a central design criterion to keep the decoder as simple as possible and to keep its computational complexity low.

5. LISTENING RESULTS

Subjective listening tests by Google[18] and Noki-

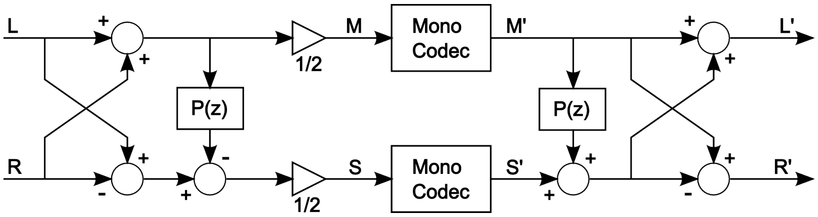

Fig. 7: Stereo prediction block diagram.

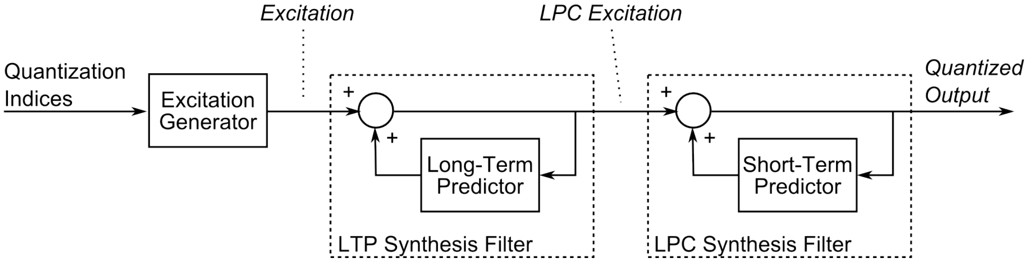

Fig. 8: Decoder side linear prediction block diagram.

a[19] show that Opus outperforms most existing speech codecs at all but the lowest bitrates.

In [18], MUSHRA-type tests were used, and the following conclusions were made for WB and FB:

- Opus at 32 kbps is better than G.719 at 32 kbps.
- Opus at 20 kbps is better than Speex and G.722.1 at 24 kbps.
- Opus at 11 kbps is better than Speex at 11 kbps.

In [19], it is stated that:

- Hybrid mode provides excellent voice quality at bitrates from 20 to 40 kbit/s.

6. CONCLUSION

We have in this paper described the voice mode in Opus. The paper is intended to complement the paper about music mode [3], for a complete description of the codec. The format of the paper makes it easier to approach than the more comprehensive RFC 6716 [2].

7. REFERENCES

[1] Opus Interactive Audio Codec, http://www.-opus-codec.org/.

[2] J.-M. Valin, K. Vos, and T. B. Terriberry, “Definition of the Opus Audio Codec” RFC 6716, http://www.ietf.org/rfc/rfc6716.txt, Amsterdam, The Netherlands, September 2012.

[3] J.-M. Valin, G Maxwell, T. B. Terriberry, and K. Vos, ”High-Quality, Low-Delay Music Coding in the Opus Codec”, Accepted at the AES 135th Convention, 2013.

[4] K. Vos, S. Jensen, and K. Sørensen, ”SILK speech codec”, IETF Internet-Draft, http://-tools.ietf.org/html/draft-vos-silk-02.

[5] Burg, J., ”Maximum Entropy Spectral Analysis”, Proceedings of the 37th Annual International SEG Meeting, Vol. 6, 1975.

[6] K. Vos, ”A Fast Implementation of Burg's Method”, www.arxiv.org, 2013.

[7] P. Kabal and R. P. Ramachandran, ”Joint Solutions for Formant and Pitch Predictors in Speech Processing”, Proc. IEEE Int. Conf. A-coustics, Speech, Signal Processing (New York, NY), pp. 315-318, April 1988.

[8] H.W. Strube, ”Linear prediction on a Warped Frequency Scale”, Journal of the Acoustical So-

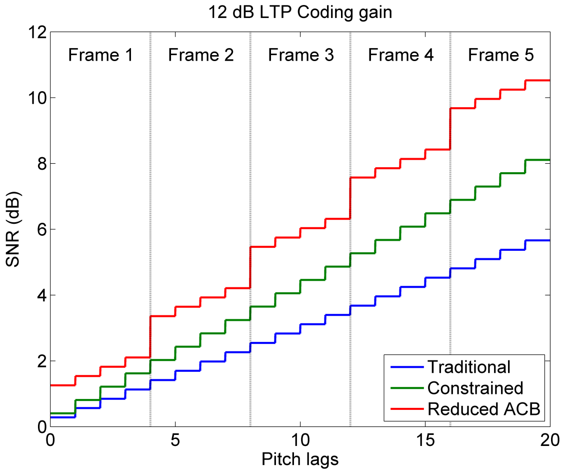

Fig. 6: Illustration of convergence speed after a packet loss by measuring the SNR of the zero state LTP filter response. The traditional solution means standard LTP. Constrained is the method in [11], where the LTP prediction gain is constrained which adds 1/4 bit per sample. Reduced ACB is the Opus method. The experiment is made with a pitch lag of 1/4 packet length, meaning that the Opus method can add 1 bit per sample in the first pitch period in order to balance the extra rate for constrained LTP. The unconstrained LTP prediction gain is set to 12 dB, and high-rate quantization theory is assumed (1 bit/sample ↔ 6 dB SNR). After 5 packets the Opus method outperforms the alternative methods by > 2 dB and the standard by 4 dB.

ciety of America, vol. 68, no. 4, pp. 10711076, Oct 1980.

[9] B. Atal and M. Schroeder, "Predictive Coding of Speech Signals and Subjective Error Criteria", IEEE Tr. on Acoustics Speech and Signal Processing, pp. 247-254, July 1979.

[10] Laroia, R., Phamdo, N., and N. Farvardin, "Robust and Efficient Quantization of Speech LSP Parameters Using Structured Vector Quantization", ICASSP-1991, Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing, pp. 641-644, October 1991.

[11] M. Chibani, P. Gournay, and R. Lefebvre, "Increasing the Robustness of CELP-Based Coders by Constrained Optimization", in Proc IEEE Int. Conf. on Acoustics, Speech and Signal Processing, March 2005.

[12] J. B. Anderson, T. Eriksson, M. Novak, Trellis source codes based on linear congruential recursions, Proc. IEEE International Symposium on Information Theory, 2003.

[13] E. Ayanoglu and R. M. Gray, "The Design of Predictive Trellis Waveform Coders Using the Generalized Lloyd Algorithm", IEEE Tr. on Communications, Vol. 34, pp. 1073-1080, November 1986.

[14] J. B. Bodie, Multi-path tree-encoding for analog data sources, Commun. Res. Lab., Fac. Eng., McMasters Univ., Hamilton, Ont., Canada, CRL Int. Rep., Series CRL-20, 1974.

[15] P. Hedelin and J. Skoglund, Vector quantization based on Gaussian mixture models, IEEE Trans. Speech and Audio Proc., vol. 8, no. 4, pp. 385401, Jul. 2000.

[16] H. Krüger and P. Vary, A New Approach for Low-Delay Joint-Stereo Coding, ITG-Fachtagung Sprachkommunikation, VDE Verlag GmbH, Oct. 2008.

[17] G. Nigel and N. Martin, Range encoding: An algorithm for removing redundancy from a digitized message, Video & Data Recording Conference, Southampton, UK, July 2427, 1979.

[18] J. Skoglund, "Listening tests of Opus at Google", IETF, 2011.

[19] A. Rämö, H. Toukomaa, "Voice Quality Characterization of IETF Opus Codec", Interspeech, 2011.