Design and Description of CS-ACELP: A Toll Quality 8 kb/s Speech Coder

Redwan Salami, *Member, IEEE,* Claude Laflamme, Jean-Pierre Adoul, *Fellow, IEEE,* Akitoshi Kataoka, *Member, IEEE,* Shinji Hayashi, Takehiro Moriya, *Member, IEEE,* Claude Lamblin, Dominique Massaloux, Stéphane Proust, Peter Kroon, *Fellow, IEEE,* and Yair Shoham, *Member, IEEE*

***Abstract*— This paper describes the 8 kb/s speech coding algorithm G.729 which has been recently standardized by ITU-T. The algorithm is based on a conjugate-structure algebraic CELP (CS-ACELP) coding technique and uses 10 ms speech frames. The codec delivers toll-quality speech (equivalent to 32 kb/s ADPCM) for most operating conditions. This paper describes the coder structure in detail and discusses the reasons behind certain design choices. A 16-b fixed-point version has been developed as part of Recommendation G.729 and a summary of the subjective test results based on a real-time implementation of this version are presented.**

***Index Terms*—Analysis-by-synthesis, speech coding.**

I. Introduction

SINCE 1990, Study Group 15 (SG15) of the ITU-T has been involved in a standardization process for a speech coding algorithm at 8 kb/s. The main applications for this coder are 1) personal communication systems (PCS), 2) digital satellite systems, and 3) other applications such as packetized speech and circuit multiplexing equipment. The speech quality produced by this coder should be equivalent to that of 32 kb/s ADPCM (G.726) for most operating conditions. These conditions include clean and noisy speech, multiple encodings, level variations and nonspeech inputs. The intended wireless applications require that the coder is robust against channel errors. These errors could be either random or bursty, and the coder should be able to withstand them without introducing major annoying effects. Moreover, if the radio channels suffer from long fades, and complete frames are lost, the decoder should be able to conceal these missing frames with a minimal loss in speech quality.

Two candidate algorithms were submitted: one from NTT [1]–[3] and the other from France Telecom CNET/University of Sherbrooke [4]. Both candidates were equivalent to (or better than) 32 kb/s ADPCM in most test conditions; however, they failed some conditions. At the March 1994 meeting of SG15, both proponents agreed to join their efforts to produce a coder that combines the best features of both algorithms, and to undertake further research to meet all performance requirements. At this time, AT&T joined these algorithmic optimization efforts. A floating-point version of the resulting coder was tested in January 1995, and it was accepted at the ITU-T meeting in February 1995. In the final recommendation the algorithm is specified in terms of 16-b fixed-point arithmetic. This version was tested in October 1995, and the recommendation was accepted for ratification in November 1995 [5].

In this paper, we describe the important aspects of the algorithm, which is referred to as conjugate-structure algebraic CELP (CS-ACELP). Additional information can be found in [6]–[10]. The complete algorithm, including ANSI-C source code, can be found in [11].

This paper is organized as follows. In Section II we describe the coding algorithm in detail. In Section III we describe features of this coder that were included to increase the robustness against transmission errors. Section IV reports on the performance, and Section V discusses implementation aspects. Finally, the conclusions are given in Section VI.

II. Description of the CS-ACELP Speech Coder

The coder is based on a code-excited linear prediction (CELP) coding model [12]. In this model the locally decoded signal is compared against the original signal and the coder parameters are selected such that the mean-squared weighted error between the original and reconstructed signal is minimized.

The CS-ACELP coder is designed to operate with an appropriately bandlimited signal sampled at 8000 Hz. The input and output samples are represented using 16-b linear PCM. The coder operates on frames of 10 ms, using a 5 ms look-ahead for linear prediction (LP) analysis. This results in an overall algorithmic delay of 15 ms. The encoding principle is shown in Fig. 1. After processing the 16-b input samples through a 140 Hz highpass filter, tenth-order LP analysis is performed, and the LP parameters are quantized in the line spectral pair (LSF) domain [13] with 18 b [7]. The input frame is divided into two subframes of 5 ms each. The use of subframes allows better tracking of the pitch and gain parameters and reduces the complexity of the codebook searches. The quantized and unquantized LP filter coefficients are used for the second subframe while in the first subframe interpolated LP filter coefficients are used. For each subframe

Manuscript received March 21, 1996; revised March 26, 1997. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. W. Bastiaan Kleijn.

R. Salami, C. Laflamme, and J.-P. Adoul are with the Department of Electrical Engineering, University of Sherbrooke, P.Q., Canada J1K 2R1.

A. Kataoka, S. Hayashi, and T. Moriya are with NTT, Tokyo, Japan.

C. Lamblin, D. Massaloux, and S. Proust are with France Telecom CNET, Lannion, France.

P. Kroon and Y. Shoham are with Bell Laboratories, Lucent Technologies, Murray Hill, NJ 07974 USA (e-mail: kroon@research.bell-labs.com).

Publisher Item Identifier S 1063-6676(98)01691-5.

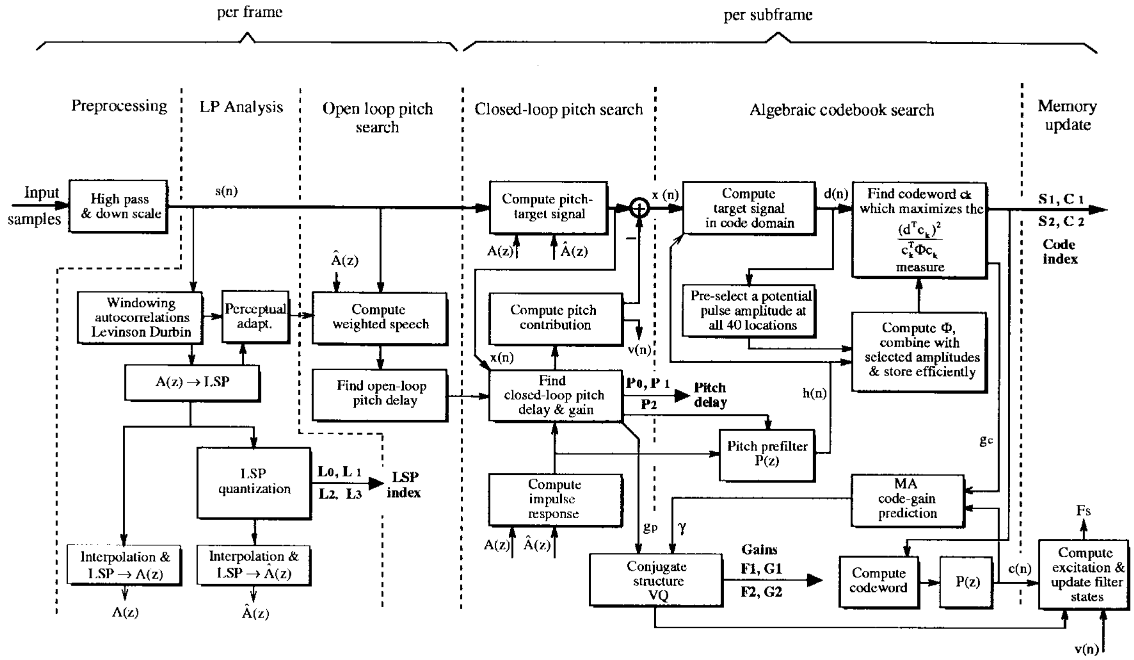

Fig. 1. Block diagram of the CS-ACELP encoder.

the excitation is represented by an adaptive-codebook and a fixed-codebook contribution. The adaptive and fixed-codebook parameters are transmitted every subframe.

The adaptive-codebook component represents the periodicity in the excitation signal using a fractional pitch lag [14] with 1/3 sample resolution. The adaptive-codebook is searched using a two-step procedure. An open-loop pitch lag is estimated once per frame based on the perceptually weighted speech signal. The adaptive-code book index and gain are found by a closed-loop search around the open-loop pitch lag. The signal to be matched, referred to as the target signal, is computed by filtering the LP residual through the weighted synthesis filter.

The adaptive-codebook index is encoded with 8 b in the first subframe and differentially encoded with 5 b in the second subframe. The target signal is updated by removing the adaptive-codebook contribution, and this new target is used in the fixed-codebook search. The fixed codebook is a 17-b algebraic codebook [10]. The gains of the adaptive and fixed codebook are vector quantized with 7 b using a conjugate-structure codebook [7] (with moving-average (MA) prediction applied to the fixed-codebook gain as in [4] and [15]). The bit allocation for a 10 ms frame is shown in Table I.

The function of the decoder (see Fig. 2) consists of decoding the transmitted parameters (LP parameters, adaptive-codebook vector, fixed-codebook vector, and gains) and performing synthesis to obtain the reconstructed speech, followed by a postprocessing stage [8], consisting of an adaptive postfilter and a fixed highpass filter.

A. Preprocessing

The 16-b PCM input samples to the speech encoder are filtered with a second-order pole/zero highpass filter with a cutoff

low-frequency or DC components. To prevent overflow in the fixed-point implementation, the input values are divided by two. The filtered and scaled signal is referred to as $s(n)$, and will be used in all subsequent encoder operations.

B. LP Analysis and Quantization

LP analysis is performed once per speech frame using the autocorrelation method [16] with a 30 ms asymmetric window. Every 80 samples (10 ms), the autocorrelation coefficients of windowed speech are computed and converted to LP coefficients using the Levinson–Durbin algorithm [16]. Then the LP coefficients are transformed to line spectral frequencies (LSF) [13] for quantization and interpolation purposes. The interpolated quantized and unquantized LSF coefficients are converted back to LP coefficients to construct the synthesis and weighting filters for each subframe. The short-term analysis and synthesis filters are based on tenth-order LP filters. The LP synthesis filter is defined as

$$\frac{1}{\hat{A}(z)} = \frac{1}{1 + \sum_{i=1}^{10} \hat{a}_i z^{-i}} \quad (1)$$

where $\hat{a}_i$, $i = 1, \cdots, 10$, are the (quantized) LP coefficients.

1) Windowing and Autocorrelation Computation: The LP analysis window consists of two parts: the first part is half a Hamming window and the second part is a quarter of a cosine function cycle. The window is given by

$$w_{\text{lp}}(n) = \begin{cases} 0.54 - 0.46 \cos\left(\frac{2\pi n}{399}\right), & n = 0, \cdots, 199, \\ \cos\left(\frac{2\pi(n-200)}{159}\right), & n = 200, \cdots, 239. \end{cases} \quad (2)$$

There is a 5 ms look-ahead in the LP analysis, which means that 40 samples are needed from the future speech frame. This translates into an extra algorithmic delay of 5 ms at the encoder stage. The use of an asymmetrical window allows reduction

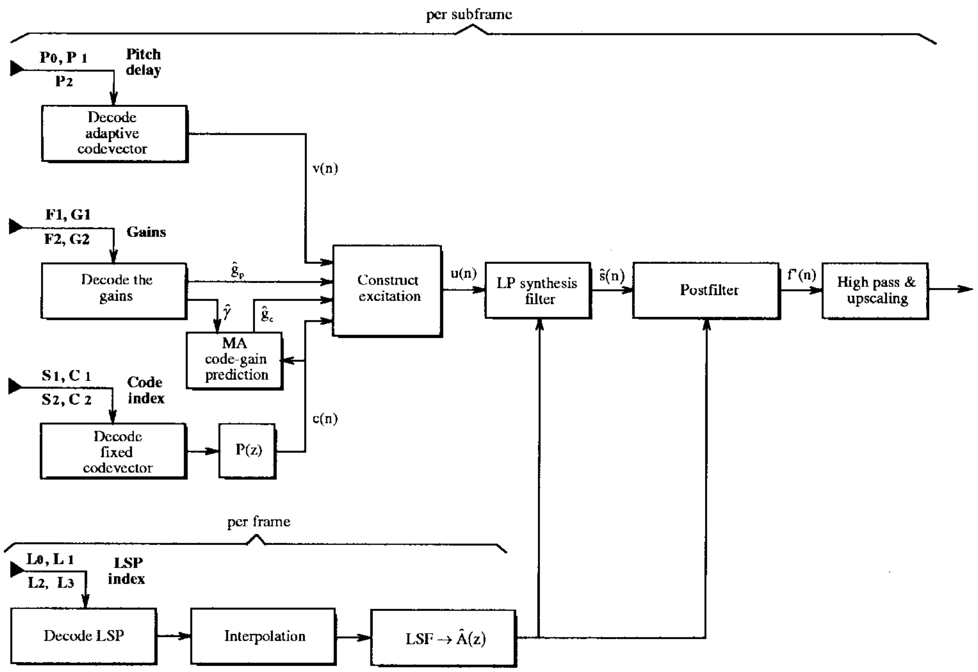

Fig. 2. Block diagram of the CS-ACELP decoder.

TABLE I
BIT ALLOCATION OF G.729 CS-ACELP FOR A 10 ms FRAME. FOR SOME PARAMETERS, THE NUMBER OF BITS FOR EACH 5 ms SUBFRAME IS IDENTIFIED. THE TOTAL NUMBER OF BITS FOR A 10 ms FRAME = 80

Parameter	*Codeword*	*Bits*
Line Spectrum Pairs	$L0, L1, L2, L3$	18
Adaptive-codebook index	$P1, P2$	8, 5
Parity for $P1$	$P0$	1
Fixed-codebook index	$C1, C2$	13, 13
Fixed-codebook pulse signs	$S1, S2$	4, 4
Codebook gains (stage 1)	$F1, F2$	3, 3
Codebook gains (stage 2)	$G1, G2$	4, 4

analysis window is applied to 120 samples from past speech frames, 80 samples from the present speech frame, and 40 samples from the future frame. The use of a 30 ms window was found to provide a smoother evolution of the LP filter, thereby providing better speech quality.

The autocorrelation coefficients are computed from the windowed speech

$$r(k) = \sum_{n=k}^{239} w_{\text{lp}}(n)s(n)w_{\text{lp}}(n-k)s(n-k), \quad k = 0, \cdots, 10. \tag{3}$$

To avoid arithmetic problems for low-level input signals the value of $r(0)$ has a lower boundary of $r(0) = 1$. A 60 Hz bandwidth expansion [18] is applied, by multiplying the autocorrelation coefficients with

$$\left[-\frac{1}{2}\left(\frac{2\pi f_0 k}{f_s}\right)^2 \right] \tag{4}$$

where $f_0 = 60$ Hz is the bandwidth expansion and $f_s = 8000$ Hz is the sampling frequency. The bandwidth expansion on the autocorrelation coefficients reduces the possibility of ill-conditioning in the Levinson algorithm (especially in fixed point). In addition, it reduces underestimation of the formant bandwidths, which could create undesirably sharp resonances.

To further reduce the possibility of ill-conditioning due to bandpass filtering of the input, the value of $r(0)$ is multiplied by a white-noise correction factor 1.0001, which is equivalent to adding a noise floor at -40 dB [19]. The modified autocorrelation coefficients are used to obtain the LP filter coefficients a_i, $i = 1, \cdots, 10$, by using the Levinson–Durbin algorithm [16].

2) Quantization of the LSF Coefficients: The LP filter coefficients $a_i, i = 1, \cdots, 10$ are converted to line spectral frequencies (LSF) using Chebyshev polynomials [20]. In this procedure the roots are found in the cosine domain. Since the quantizer is vector quantization (VQ) based, it is more convenient to represent the LSF's as normalized radian frequencies. The relation between these two representations is given by

$$\omega_i = \arccos(q_i), \quad i = 1, \cdots, 10 \tag{5}$$

where q_i are the LSF coefficients in the cosine domain, and ω_i the LSF coefficients in the frequency domain.

To keep the algorithmic delay as low as possible, the update of the LP coefficients is done every 10 ms. However, most speech spectra vary slowly in time, and a slower update (e.g., 20 ms) would provide a better tradeoff between spectral representation and bit rate. Since the higher update rate introduces a strong correlation between coefficients from frame to frame, a good compromise is to use predictive VQ. During onsets this correlation is not very strong. To accommodate both types of correlation the predictor switches between two modes mild correlation. Another advantage of using a separate bit for the switch is that it effectively reduces the size of the codebook, thereby reducing the storage requirements. To limit propagation of channel errors, the predictor is based on an MA filter. The length of this filter was determined empirically using a large data base [2], and it was found that a fourth-order MA predictor forms a good compromise between performance and error propagation. The quantizer is organized as follows: a switched fourth-order MA prediction is used to predict the LSF coefficients of the current frame. The difference between the computed and predicted coefficients is quantized using a two-stage vector quantizer. The first stage is a 10-D VQ using codebook $\mathcal{L}1$ with 128 entries (7 b). The second stage is a 10-b VQ that has been implemented as a split VQ using two 5-D codebooks, $\mathcal{L}2$ and $\mathcal{L}3$ containing 32 entries (5 b) each. The reason for using a nonsplit first-stage is that it allows for the exploitation of the correlations between the first 5 LSF and last 5 LSF coefficients. At the second stage these correlations are less strong, and the split reduces search time and storage requirements.

To explain the quantization process, it is convenient to first describe the decoding process. Each quantized value is obtained from the sum of two codewords, as follows:

$$\hat{l}_i = \begin{cases} \mathcal{L}1_i(L1) + \mathcal{L}2_i(L2), & i = 1, \cdots, 5, \\ \mathcal{L}1_i(L1) + \mathcal{L}3_{i-5}(L3), & i = 6, \cdots, 10 \end{cases} \tag{6}$$

where $L1$, $L2$, and $L3$ are the codebook indices. To guarantee that the reconstructed filters are stable the vector $\hat{l}_i$ is arranged such that adjacent elements have a minimum distance of $d_{\min}$ (see [11]). This rearrangement process is done twice. First with a value of $d_{\min} = 0.0012$, then with a value of $d_{\min} = 0.0006$. The incorporation of this process into the quantization procedure, assures that each of the possible reconstructed l_i vectors produces a stable filter. After this rearrangement process, the quantized LSF coefficients $\hat{\omega}_i^{(m)}$ for the current frame m, are obtained from the weighted sum of previous quantizer outputs $\hat{l}_i^{(m-k)}$, and the current quantizer output $\hat{l}_i^{(m)}$

$$\hat{\omega}_i^{(m)} = \left(1 - \sum_{k=1}^{4} \hat{p}_{i,k}\right)\hat{l}_i^{(m)} + \sum_{k=1}^{4} \hat{p}_{i,k}\hat{l}_i^{(m-k)}, \quad i = 1, \cdots, 10 \tag{7}$$

where $\hat{p}_{i,k}$ are the coefficients of the switched MA predictor as defined by codebook $\mathcal{L}0$, and the one bit codebook index $L0$. At startup the initial values of $\hat{l}_i^{(k)}$ are given by $\hat{l}_i = i\pi/11$ for all $k < 0$.

After computing $\hat{\omega}_i$, the corresponding filter is checked for stability and unnatural sharp resonances by checking the ordering property (i.e., $0 < \omega_1 < \omega_2 < \cdots < \omega_{10} < \pi$). If this condition is not met, the frequencies are moved using a heuristic process, which enforces a minimum spacing of 50 Hz between the coefficients [11].

The procedure for encoding the LSF parameters can be outlined as follows. For each of the two MA predictors the best approximation to the current LSF coefficients has to be found,

the weighted mean-squared error

$$E_{\text{lsf}} = \sum_{i=1}^{10} w_i(\omega_i - \hat{\omega}_i)^2. \tag{8}$$

The weights emphasize the relative importance of each LSF. Spectral resonances (closely spaced LSF's) are perceptually more important, and several (heuristic) procedures to derive these coefficients can be found in the literature (cf. [21]). The following heuristic procedure was found to improve performance and be computationally efficient. The weights w_i are made adaptive as a function of the unquantized LSF coefficients,

$$\begin{aligned} w_1 &= \begin{cases} 1, & \text{if } \omega_2 - 0.04\pi - 1 > 0, \\ 10(\omega_2 - 0.04\pi - 1)^2 + 1, & \text{otherwise} \end{cases} \\ w_{2,\cdots,9} &= \begin{cases} 1, & \text{if } \omega_{i+1} - \omega_{i-1} - 1 > 0, \\ 10(\omega_{i+1} - \omega_{i-1} - 1)^2 + 1, & \text{otherwise} \end{cases} \\ w_{10} &= \begin{cases} 1, & \text{if } -\omega_9 + 0.92\pi - 1 > 0, \\ 10(-\omega_9 + 0.92\pi - 1)^2 + 1, & \text{otherwise.} \end{cases} \end{aligned} \tag{9}$$

In addition, the weights w_5 and w_6 are multiplied by 1.2 each.

The vector to be quantized for the current frame m is obtained from

$$l_i^{(m)} = \left[\omega_i^{(m)} - \sum_{k=1}^{4} \hat{p}_{i,k}\hat{l}_i^{(m-k)}\right] \Bigg/ \left(1 - \sum_{k=1}^{4} \hat{p}_{i,k}\right), \quad i = 1, \cdots, 10. \tag{10}$$

The first codebook $\mathcal{L}1$ is searched and the entry $L1$ that minimizes the (unweighted) mean-squared error (MSE) is selected. This is followed by a search of the second codebook $\mathcal{L}2$, which defines the lower part of the second stage. The weighted MSE of (8) is computed, and the vector $L2$ which results in the lowest error is selected. Using the selected first stage vector $L1$ and the lower part of the second stage $L2$, the higher part of the second stage is searched from codebook $\mathcal{L}3$. The vector $L3$ that minimizes the weighted MSE is selected. The resulting vector $\hat{l}_i, i = 1, \cdots, 10$ is rearranged twice using the procedure outlined earlier. This process is done for each of the two MA predictors defined by $\mathcal{L}0$, and the MA predictor $L0$ that produces the lowest weighted MSE is selected.

3) Interpolation of the LSF Coefficients: The quantized (and unquantized) LP coefficients are used for the second subframe. For the first subframe, the quantized (and unquantized) LP coefficients are obtained by linear interpolation of the corresponding parameters in the adjacent subframes. The interpolation is done on the LSF coefficients in the cosine domain rather than the frequency domain. Interpolating in either domain did not produce noticeable audible differences, and the cosine domain was selected because of ease of implementation. Once the LSF coefficients are quantized and interpolated, they are converted back to the LP coefficients $\hat{a}_i$.

C. Perceptual Weighting

The weighted speech signal $s_w(n)$ in a subframe is obtained

$W(z)$. This perceptual weighting filter [22] is based on the unquantized LP filter coefficients a_i, and is given by

$$W(z) = \frac{A(z/\gamma_1)}{A(z/\gamma_2)} = \frac{1+\sum_{i=1}^{10}\gamma_1^i a_i z^{-i}}{1+\sum_{i=1}^{10}\gamma_2^i a_i z^{-i}}. \qquad (11)$$

The use of the unquantized coefficients gives a weighting filter that matches better the original spectrum. The values of γ_1 and γ_2 modify the frequency response of the filter $W(z)$, and thereby the amount of noise weighting. It is difficult to find fixed values of γ_1 and γ_2 that provided good performance for different input signal characteristics. For example, differences in the low-frequency cutoff would lead to different choices for these coefficients. Hence, the values of γ_1 and γ_2 are made a function of the spectral shape of the input signal. For signals with a lot of low-frequency energy, the amount of weighting is increased.

This adaptation is done once per 10 ms frame, but an interpolation procedure for each first subframe is used to smooth this adaptation process. The spectral shape is obtained from a second-order linear prediction filter, obtained as a by-product from the Levinson–Durbin recursion. The reflection coefficients k_i are converted to log area ratio (LAR) coefficients o_i by

$$o_i = \log\frac{(1+k_i)}{(1-k_i)} \quad i = 1, 2. \qquad (12)$$

The LAR coefficients are used because they have better interpolation properties than reflection coefficients [23]. The LAR coefficients corresponding to the current 10 ms frame are used for the second subframe. The LAR coefficients for the first subframe are obtained through linear interpolation with the LAR parameters from the previous frame. The spectral envelope is characterized as being either flat $(S = 1)$ or tilted $(S = 0)$. For each subframe this characterization is obtained by applying a threshold function to the LAR coefficients. To avoid rapid changes, a hysteresis is used by taking into account the value of S in the previous subframe $m - 1$,

$$S^{(m)} = \begin{cases} 0, & \text{if } o_1^{(m)} < -1.74 \text{ and } o_2^{(m)} > 0.65 \text{ and } S^{(m-1)} = 1, \\ 1, & \text{if } (o_1^{(m)} > -1.52 \text{ or } o_2^{(m)} < 0.43) \text{ and } S^{(m-1)} = 0, \\ S^{(m-1)}, & \text{otherwise} \end{cases} . \qquad (13)$$

If the interpolated spectrum for a subframe is classified as flat $(S^{(m)} = 1)$, the weight factors are set to $\gamma_1 = 0.94$ and $\gamma_2 = 0.6$. If the spectrum is classified as tilted $(S^{(m)} = 0)$, the value of γ_1 is set to 0.98, and the value of γ_2 is adapted to the strength of the resonances in the LP synthesis filter, but is bounded between 0.4 and 0.7. If a strong resonance is present, the value of γ_2 is set closer to the upperbound. This adaptation is done to reduce the amount of unmasked noise at the formant frequencies. The adaptation of γ_2 is done by using a heuristic criterion based on the minimum distance between two successive LSF coefficients for the current subframe. The minimum distance is given by

The value of γ_2 is computed using

$$\gamma_2 = -6d_{\min} + 1, \text{ bounded by } 0.4 \le \gamma_2 \le 0.7. \qquad (15)$$

The values of γ_1 and γ_2 for the different conditions were obtained through many informal listening experiments using expert listeners. The process was done in stages by selecting speech material that could be characterized to fall into one of the categories: 1) flat spectrum $(S = 1)$, 2) tilted spectrum $(S = 0)$ and no strong resonances, and 3) tilted spectrum $(S = 0)$ with strong resonances. This allowed independent optimization of the weight factors. In all experiments both single and double encodings were considered. In general, the improvements due to this adaptation of the weights are most noticeable for double encodings.

D. Pitch Analysis

The pitch analysis technique described in [4] is used. An open-loop pitch lag T_{op} is estimated once per 10 ms frame using the weighted speech signal $s_w(n)$. The adaptive-codebook approach is used to represent the periodic component in the excitation signal. The selected adaptive-codebook vector is represented by an index, which corresponds to a certain fractional lag value.

For each subframe the target signal, $x(n)$, and the impulse response, $h(n)$, of the weighted synthesis filter are computed. A closed-loop adaptive-codebook search is performed in the first subframe around the index corresponding to the open-loop pitch lag estimate (± 3). A 1/3 fractional sample resolution is used in the range $[19\frac{1}{3}, 84\frac{2}{3}]$ and integers only are used in the range 85–143. It was found that this choice of resolution provides a good trade-off between performance and bit rate. The adaptive-codebook index in the first subframe is encoded with 8 b. In the second subframe, a 1/3 fractional sample resolution is used in the range $[T_1 - 5\frac{2}{3}, T_1 + 4\frac{2}{3}]$ where T_1 is the integer part of the adaptive-codebook lag in the first subframe. This range is adapted for the cases where T_1 straddles the boundaries of the lag range. The lag in the second subframe is differentially encoded with 5 b. Since the open-loop pitch estimate provides a form of pitch tracking, the differential coding does not introduce noticeable degradations in the speech quality.

1) Open-Loop Pitch Lag Estimation: The open-loop pitch lag estimation uses the weighted speech signal $s_w(n)$, and is done as follows: in the first step, 3 maxima of the correlation

$$R(k) = \sum_{n=0}^{79} s_w(n)s_w(n-k) \qquad (16)$$

are found in the following three ranges: 1) $k = 80, \ldots, 143$, 2) $k = 40, \cdots, 79$, and 3) $k = 20, \cdots, 39$. Note that for $n-k < 0$ signal values from the previous frame are used. The retained maxima $R(t_i)$, where t_i are the lag values corresponding to the maxima in the three lag regions $i = 1, \cdots, 3$, are normalized through

$$R'(t_i) = \frac{R(t_i)}{\ldots} \quad i = 1, \ldots, 3. \qquad (17)$$

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS

Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS

Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS

Sync your system to PACER to automate legal marketing.

WHAT WILL YOU BUILD? | **sales@docketalarm.com** | **1-866-77-FASTCASE**

fastcase®
Smarter legal research.