3GPP TS 26.190 V1.0.0 (2000-12)

Technical Specification

3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Mandatory Speech Codec speech processing functions; AMR Wideband speech codec; Transcoding functions (Release 4)

Keywords
AMR, CODEC, Adaptive Multi-Rate, Wideband speech coder

3GPP

Postal address

3GPP support office address
650 Route des Lucioles - Sophia Antipolis
Valbonne - FRANCE
Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Internet
http://www.3gpp.org

Contents

Foreword 5
1 Scope 6
2 Normative references 6
3 Definitions, symbols and abbreviations 7
3.1 Definitions 7
3.2 Symbols 8
3.3 Abbreviations 13
4 Outline description 13
4.1 Functional description of audio parts 13
4.2 Preparation of speech samples 14
4.3 Principles of the adaptive multi-rate wideband speech encoder 14
4.4 Principles of the adaptive multi-rate speech decoder 16
4.5 Sequence and subjective importance of encoded parameters 17
5 Functional description of the encoder 17
5.1 Pre-processing 17
5.2 Linear prediction analysis and quantization 18
5.2.1 Windowing and auto-correlation computation 18
5.2.2 Levinson-Durbin algorithm 18
5.2.3 LP to ISP conversion 19
5.2.4 ISP to LP conversion 20
5.2.5 Quantization of the ISP coefficients 21
5.2.6 Interpolation of the ISPs 22
5.3 Perceptual weighting 22
5.4 Open-loop pitch analysis 23
5.4.1 6.60 kbit/s mode 23
5.4.2 8.85, 12.65, 14.25, 15.85, 18.25, 19.85, 23.05 or 23.85 kbit/s modes 24
5.5 Impulse response computation 24
5.6 Target signal computation 24
5.7 Adaptive codebook 25
5.8 Algebraic codebook 26
5.8.1 Codebook structure 26
5.8.1.1 23.85 and 23.05 kbit/s mode 27
5.8.1.2 19.85 kbit/s mode 27
5.8.1.3 18.25 kbit/s mode 27
5.8.1.4 15.85 kbit/s mode 28
5.8.1.5 14.25 kbit/s mode 28
5.8.1.6 12.65 kbit/s mode 28
5.8.1.7 8.85 kbit/s mode 29
5.8.1.8 6.60 kbit/s mode 29
5.8.2 Pulse indexing 29
5.8.3 Codebook search 33
5.9 Quantization of the adaptive and fixed codebook gains 36
5.10 Memory update 37
5.11 High-band gain generation 37
6 Functional description of the decoder 38
6.1 Decoding and speech synthesis 38
6.2 High-pass filtering, up-scaling and interpolation 41
6.3 High frequency band 41
6.3.1 Generation of high-band excitation 41
6.3.2 LP filter for the high frequency band 42
6.3.2.1 6.60 kbit/s mode 42
6.3.2.2 8.85, 12.65, 14.25, 15.85, 18.25, 19.85, 23.05 or 23.85 kbit/s modes 42
6.3.3 High band synthesis 42

7 Detailed bit allocation of the adaptive multi-rate wideband codec 44
8 Bibliography 51

Annex A (informative): Change history 52

Foreword

The present document describes the detailed mapping of the wideband telephony speech service employing the Adaptive Multi-Rate (AMR-WB) speech coder within the 3GPP system.

The contents of the present document are subject to continuing work within the TSG and may change following formal TSG approval. Should the TSG modify the contents of this TS, it will be re-released by the TSG with an identifying change of release date and an increase in version number as follows:

Version x.y.z

where:

- x the first digit:
  - 1 presented to TSG for information;
  - 2 presented to TSG for approval;
  - 3 Indicates TSG approved document under change control.
- y the second digit is incremented for all changes of substance, i.e. technical enhancements, corrections, updates, etc.
- z the third digit is incremented when editorial only changes have been incorporated in the specification;

1 Scope

This Telecommunication Standard (TS) describes the detailed mapping from input blocks of 320 speech samples in 16-bit uniform PCM format to encoded blocks of 132, 177, 253, 285, 317, 365, 397, 461 and 477 bits and from encoded blocks of 132, 177, 253, 285, 317, 365, 397, 461 and 477 bits to output blocks of 320 reconstructed speech samples. The sampling rate is 16 000 samples/s leading to a bit rate for the encoded bit stream of 6.60, 8.85, 12.65, 14.25, 15.85, 18.25, 19.85, 23.05 or 23.85 kbit/s. The coding scheme for the multi-rate coding modes is the so-called Algebraic Code Excited Linear Prediction Coder, hereafter referred to as ACELP. The wideband multi-rate ACELP coder is referred to as MRWB-ACELP.

2 Normative references

This TS incorporates by dated and undated reference, provisions from other publications. These normative references are cited in the appropriate places in the text and the publications are listed hereafter. For dated references, subsequent amendments to or revisions of any of these publications apply to this TS only when incorporated in it by amendment or revision. For undated references, the latest edition of the publication referred to applies.

[1] GSM 03.50: " Digital cellular telecommunications system (Phase 2); Transmission planning aspects of the speech service in the GSM Public Land Mobile Network (PLMN) system"

[2] 3G TS 26.201 : "AMR wideband speech codec; Frame structure".

[3] 3G TS 26. 194: "AMR wideband speech codec; Voice Activity Detection (VAD)".

[4] 3G TS 26. 173: "AMR wideband speech codec; ANSI-C code".

[5] 3G TS 26. 174: "AMR wideband speech codec; Test sequences".

[6] ITU-T Recommendation G.711 (1988): "Coding of analogue signals by pulse code modulation Pulse code modulation (PCM) of voice frequencies".

3 Definitions, symbols and abbreviations

3.1 Definitions

For the purposes of this TS, the following definitions apply:

adaptive codebook: The adaptive codebook contains excitation vectors that are adapted for every subframe. The adaptive codebook is derived from the long-term filter state. The lag value can be viewed as an index into the adaptive codebook.

algebraic codebook: A fixed codebook where algebraic code is used to populate the excitation vectors (innovation vectors). The excitation contains a small number of nonzero pulses with predefined interlaced sets of potential positions. The amplitudes and positions of the pulses of the k^{th} excitation codevector can be derived from its index k through a rule requiring no or minimal physical storage, in contrast with stochastic codebooks whereby the path from the index to the associated codevector involves look-up tables.

anti-sparseness processing: An adaptive post-processing procedure applied to the fixed codebook vector in order to reduce perceptual artifacts from a sparse fixed codebook vector.

closed-loop pitch analysis: This is the adaptive codebook search, i.e., a process of estimating the pitch (lag) value from the weighted input speech and the long term filter state. In the closed-loop search, the lag is searched using error minimization loop (analysis-by-synthesis). In the adaptive multi-rate codec, closed-loop pitch search is performed for every subframe.

direct form coefficients: One of the formats for storing the short term filter parameters. In the adaptive multi-rate codec, all filters which are used to modify speech samples use direct form coefficients.

fixed codebook: The fixed codebook contains excitation vectors for speech synthesis filters. The contents of the codebook are non-adaptive (i.e., fixed). In the adaptive multi-rate codec, the fixed codebook is implemented using an algebraic codebook.

fractional lags: A set of lag values having sub-sample resolution. In the adaptive multi-rate codec a sub-sample resolution of 1/4th or 1/2nd of a sample is used.

frame: A time interval equal to 20 ms (320 samples at an 16 kHz sampling rate).

Immittance Spectral Frequencies: (see Immittance Spectral Pair)

Immittance Spectral Pair: Transformation of LPC parameters. Immittance Spectral Pairs are obtained by decomposing the inverse filter transfer function A(z) to a set of two transfer functions, one having even symmetry and the other having odd symmetry. The Immittance Spectral Pairs (also called as Immittance Spectral Frequencies) are the roots of these polynomials on the z-unit circle.

integer lags: A set of lag values having whole sample resolution.

interpolating filter: An FIR filter used to produce an estimate of sub-sample resolution samples, given an input sampled with integer sample resolution. In this implementation, the interpolating filter has low pass filter characteristics. Thus the adaptive codebook consists of the low-pass filtered interpolated past excitation.

inverse filter: This filter removes the short term correlation from the speech signal. The filter models an inverse frequency response of the vocal tract.

lag: The long term filter delay. This is typically the true pitch period, or its multiple or sub-multiple.

LP analysis window: For each frame, the short term filter coefficients are computed using the high pass filtered speech samples within the analysis window. In the adaptive multi-rate codec, the length of the analysis window is always 384 samples. For all the modes, a single asymmetric window is used to generate a single set of LP coefficients. The 5 ms look-ahead is used in the analysis.

LP coefficients: Linear Prediction (LP) coefficients (also referred as Linear Predictive Coding (LPC) coefficients) is a generic descriptive term for the short term filter coefficients.

mode: When used alone, refers to the source codec mode, i.e., to one of the source codecs employed in the AMR-WB codec.

open-loop pitch search: A process of estimating the near optimal lag directly from the weighted speech input. This is done to simplify the pitch analysis and confine the closed-loop pitch search to a small number of lags around the open-loop estimated lags. In the adaptive multi-rate codec, an open-loop pitch search is performed in every other subframe.

residual: The output signal resulting from an inverse filtering operation.

short term synthesis filter: This filter introduces, into the excitation signal, short term correlation which models the impulse response of the vocal tract.

perceptual weighting filter: This filter is employed in the analysis-by-synthesis search of the codebooks. The filter exploits the noise masking properties of the formants (vocal tract resonances) by weighting the error less in regions near the formant frequencies and more in regions away from them.

subframe: A time interval equal to 5 ms (80 samples at 16 kHz sampling rate).

vector quantization: A method of grouping several parameters into a vector and quantizing them simultaneously.

zero input response: The output of a filter due to past inputs, i.e. due to the present state of the filter, given that an input of zeros is applied.

zero state response: The output of a filter due to the present input, given that no past inputs have been applied, i.e., given that the state information in the filter is all zeroes.

3.2 Symbols

For the purposes of this TS, the following symbols apply:

$A(z)$ The inverse filter with unquantized coefficients

$\hat{A}(z)$ The inverse filter with quantized coefficients

$H(z) = \frac{1}{\hat{A}(z)}$ The speech synthesis filter with quantized coefficients

a_i The unquantized linear prediction parameters (direct form coefficients)

$\hat{a}_i$ The quantified linear prediction parameters

m The order of the LP model

$W(z)$ The perceptual weighting filter (unquantized coefficients)

γ_1, γ_2 The perceptual weighting factors

T The integer pitch lag nearest to the closed-loop fractional pitch lag of the subframe

β The adaptive pre-filter coefficient (the quantified pitch gain)

$H_{h1}(z)$	Pre-processing high-pass filter
$w(n)$	LP analysis window
L_1	Length of the first part of the LP analysis window $w(n)$
L_2	Length of the second part of the LP analysis window $w(n)$
$r(k)$	The auto-correlations of the windowed speech $s'(n)$
$w_{lag}(i)$	Lag window for the auto-correlations (60 Hz bandwidth expansion)
f_0	The bandwidth expansion in Hz
f_s	The sampling frequency in Hz
$r'(k)$	The modified (bandwidth expanded) auto-correlations
$E(i)$	The prediction error in the ith iteration of the Levinson algorithm
k_i	The ith reflection coefficient
$a_j^{(i)}$	The jth direct form coefficient in the ith iteration of the Levinson algorithm
$F_1'(z)$	Symmetric ISF polynomial
$F_2'(z)$	Antisymmetric ISF polynomial
$F_1(z)$	Polynomial $F_1'(z)$
$F_2(z)$	Polynomial $F_2'(z)$ with roots $z=1$ and $z=-1$ eliminated
q_i	The immittance spectral pairs (ISPs) in the cosine domain
$\mathbf{q}$	An ISP vector in the cosine domain
$\hat{\mathbf{q}}_i^{(n)}$	The quantified ISP vector at the ith subframe of the frame n
ω_i	The immittance spectral frequencies (ISFs)
$T_m(x)$	A mth order Chebyshev polynomial
$f_1(i), f_2(i)$	The coefficients of the polynomials $F_1(z)$ and $F_2(z)$
$f_1'(i), f_2'(i)$	The coefficients of the polynomials $F_1'(z)$ and $F_2'(z)$
$f(i)$	The coefficients of either $F_1(z)$ or $F_2(z)$
$C(x)$	Sum polynomial of the Chebyshev polynomials
x	Cosine of angular frequency ω
λ_k	Recursion coefficients for the Chebyshev polynomial evaluation

f_i The immittance spectral frequencies (ISFs) in Hz

$\mathbf{f}^t = [f_1\ f_2 \ldots f_{10}]$ The vector representation of the ISFs in Hz

$\mathbf{z}(n)$ The mean-removed ISF vector at frame n

$\mathbf{r}(n)$ The ISF prediction residual vector at frame n

$\mathbf{p}(n)$ The predicted ISF vector at frame n

$\hat{\mathbf{r}}(n-1)$ The quantified residual vector at the past frame

$\hat{\mathbf{r}}_i^k$ The quantified ISF subvector i at quantization index k

d_i The distance between the immittance spectral frequencies f_{i+1} and f_{i-1}

$h(n)$ The impulse response of the weighted synthesis filter

$H(z)W(z)$ The weighted synthesis filter

T_1 The integer nearest to the fractional pitch lag of the previous (1st or 3rd) subframe

$s'(n)$ The windowed speech signal

$s_w(n)$ The weighted speech signal

$\hat{s}(n)$ Reconstructed speech signal

$x(n)$ The target signal for adaptive codebook search

$x_2(n)$, $\mathbf{x}_2^t$ The target signal for algebraic codebook search

$res_{LP}(n)$ The LP residual signal

$c(n)$ The fixed codebook vector

$v(n)$ The adaptive codebook vector

$y(n) = v(n) * h(n)$ The filtered adaptive codebook vector

$y_k(n)$ The past filtered excitation

$u(n)$ The excitation signal

$\hat{u}'(n)$ The gain-scaled emphasized excitation signal

T_{op} The best open-loop lag

t_{min} Minimum lag search value

t_{max} Maximum lag search value

$R(k)$ Correlation term to be maximized in the adaptive codebook search

$R(k)_t$	The interpolated value of $R(k)$ for the integer delay k and fraction t
A_k	Correlation term to be maximized in the algebraic codebook search at index k
C_k	The correlation in the numerator of A_k at index k
E_{Dk}	The energy in the denominator of A_k at index k
$\mathbf{d} = \mathbf{H}^t\mathbf{x}_2$	The correlation between the target signal $x_2(n)$ and the impulse response $h(n)$, i.e., backward filtered target
$\mathbf{H}$	The lower triangular Toepliz convolution matrix with diagonal $h(0)$ and lower diagonals $h(1),\ldots,h(63)$
$\mathbf{\Phi}=\mathbf{H}^t\mathbf{H}$	The matrix of correlations of $h(n)$
$d(n)$	The elements of the vector $\mathbf{d}$
$\phi(i,j)$	The elements of the symmetric matrix $\mathbf{\Phi}$
$\mathbf{c}_k$	The innovation vector
C	The correlation in the numerator of A_k
m_i	The position of the ith pulse
ϑ_i	The amplitude of the ith pulse
N_p	The number of pulses in the fixed codebook excitation
E_D	The energy in the denominator of A_k
$res_{LTP}(n)$	The normalized long-term prediction residual
$b(n)$	The signal used for presetting the signs in algebraic codebook search
$s_b(n)$	The sign signal for the algebraic codebook search
$d'(n)$	Sign extended backward filtered target
$\phi'(i,j)$	The modified elements of the matrix $\mathbf{\Phi}$, including sign information
$\mathbf{z}^t$, $z(n)$	The fixed codebook vector convolved with $h(n)$
$E(n)$	The mean-removed innovation energy (in dB)
$\overline{E}$	The mean of the innovation energy
$\tilde{E}(n)$	The predicted energy
$[b_1\ b_2\ b_3\ b_4]$	The MA prediction coefficients
$\hat{R}(k)$	The quantified prediction error at subframe k

E_I	The mean innovation energy
$R(n)$	The prediction error of the fixed-codebook gain quantization
E_Q	The quantization error of the fixed-codebook gain quantization
$e(n)$	The states of the synthesis filter $1/\hat{A}(z)$
$e_w(n)$	The perceptually weighted error of the analysis-by-synthesis search
η	The gain scaling factor for the emphasized excitation
g_c	The fixed-codebook gain
g_c'	The predicted fixed-codebook gain
$\hat{g}_c$	The quantified fixed codebook gain
g_p	The adaptive codebook gain
$\hat{g}_p$	The quantified adaptive codebook gain
$\gamma_{gc} = g_c / g_c'$	A correction factor between the gain g_c and the estimated one g_c'
$\hat{\gamma}_{gc}$	The optimum value for γ_{gc}
γ_{sc}	Gain scaling factor

3.3 Abbreviations

For the purposes of this TS, the following abbreviations apply.

ACELP	Algebraic Code Excited Linear Prediction
AGC	Adaptive Gain Control
AMR	Adaptive Multi-Rate
AMR-WB	Adaptive Multi-Rate Wideband
CELP	Code Excited Linear Prediction
FIR	Finite Impulse Response
ISF	Immittance Spectral Frequency
ISP	Immittance Spectral Pair
ISPP	Interleaved Single-Pulse Permutation
LP	Linear Prediction
LPC	Linear Predictive Coding
LTP	Long Term Predictor (or Long Term Prediction)
MA	Moving Average
S-MSVQ	Split-MultiStage Vector Quantization
WB	Wideband

4 Outline description

This TS is structured as follows:

Section 4.1 contains a functional description of the audio parts including the A/D and D/A functions. Section 4.2 describes input format for the AMR-WB encoder and the output format for the AMR-WB decoder. Sections 4.3 and 4.4 present a simplified description of the principles of the AMR-WB codec encoding and decoding process respectively. In subclause 4.5, the sequence and subjective importance of encoded parameters are given.

Section 5 presents the functional description of the AMR-WB codec encoding, whereas clause 6 describes the decoding procedures. In section 7, the detailed bit allocation of the AMR-WB codec is tabulated.

4.1 Functional description of audio parts

The analogue-to-digital and digital-to-analogue conversion will in principle comprise the following elements:

1) Analogue to uniform digital PCM

   – microphone;

   – input level adjustment device;

   – input anti-aliasing filter;

   – sample-hold device sampling at 16 kHz;

   – analogue–to–uniform digital conversion to 16–bit representation.

   The uniform format shall be represented in two's complement.

2) Uniform digital PCM to analogue

   – conversion from 16–bit/16 kHz uniform PCM to analogue;

- a hold device;
- reconstruction filter including x/sin(x) correction;
- output level adjustment device;
- earphone or loudspeaker.

In the terminal equipment, the A/D function may be achieved

- by direct conversion to 16-bit uniform PCM format;

For the D/A operation, the inverse operations take place.

4.2 Preparation of speech samples

The encoder is fed with data comprising of samples with a resolution of 16 bits left justified in a 16-bit word. The decoder outputs data in the same format. Outside the speech codec further processing must be applied if the traffic data occurs in a different representation.

4.3 Principles of the adaptive multi-rate wideband speech encoder

The AMR-WB codec consists of nine source codecs with bit-rates of 23.85 23.05, 19.85, 18.25, 15.85, 14.25, 12.65, 8.85 and 6.60 kbit/s.

The codec is based on the code-excited linear predictive (CELP) coding model. The input signal is pre-emphasized using the filter $H_{pre\text{-}emph}(z)=1-\mu z^{-1}$. The CELP model is then applied to the pre-emphasized signal. A 16th order linear prediction (LP), or short-term, synthesis filter is used which is given by:

$$H(z) = \frac{1}{\hat{A}(z)} = \frac{1}{1+\sum_{i=1}^{m} \hat{a}_i z^{-i}}, \qquad (1)$$

where $\hat{a}_i, i=1,\ldots,m$ are the (quantized) linear prediction (LP) parameters, and m = 16 is the predictor order. The long-term, or pitch, synthesis filter is usually given by:

$$\frac{1}{B(z)} = \frac{1}{1-g_p z^{-T}}, \qquad (2)$$

where T is the pitch delay and g_p is the pitch gain. The pitch synthesis filter is implemented using the so-called adaptive codebook approach.

The CELP speech synthesis model is shown in Figure 1. In this model, the excitation signal at the input of the short-term LP synthesis filter is constructed by adding two excitation vectors from adaptive and fixed (innovative) codebooks. The speech is synthesized by feeding the two properly chosen vectors from these codebooks through the short-term synthesis filter. The optimum excitation sequence in a codebook is chosen using an analysis-by-synthesis search procedure in which the error between the original and synthesized speech is minimized according to a perceptually weighted distortion measure.

The perceptual weighting filter used in the analysis-by-synthesis search technique is given by:

$$W(z) = A(z/\gamma_1)H_{de-emph}(z), \qquad (3)$$

where $A(z)$ is the unquantized LP filter, $H_{de-emph} = \frac{1}{1-0.68z^{-1}}$, and $\gamma_1=0.92$ is the perceptual weighting factor. The weighting filter uses the unquantized LP parameters.

The encoder performs the analysis of the LPC, LTP and fixed codebook parameters at 12.8 kHz sampling rate. The coder operates on speech frames of 20 ms. At each frame, the speech signal is analysed to extract the parameters of the

CELP model (LP filter coefficients, adaptive and fixed codebooks' indices and gains). In addition to these parameters, high-band gain indices are computed in 23.85 kbit/s mode. These parameters are encoded and transmitted. At the decoder, these parameters are decoded and speech is synthesized by filtering the reconstructed excitation signal through the LP synthesis filter.

The signal flow at the encoder is shown in Figure 2. After decimation, high-pass and pre-emphasis filtering is performed. LP analysis is performed once per frame. The set of LP parameters is converted to immittance spectrum pairs (ISP) and vector quantized using split-multistage vector quantization (S-MSVQ). The speech frame is divided into 4 subframes of 5 ms each (64 samples). The adaptive and fixed codebook parameters are transmitted every subframe. The quantized and unquantized LP parameters or their interpolated versions are used depending on the subframe. An open-loop pitch lag is estimated in every other subframe based on the perceptually weighted speech signal.

Then the following operations are repeated for each subframe:

- The target signal *x(n)* is computed by filtering the LP residual through the weighted synthesis filter $W(z)H(z)$ with the initial states of the filters having been updated by filtering the error between LP residual and excitation (this is equivalent to the common approach of subtracting the zero input response of the weighted synthesis filter from the weighted speech signal).

- The impulse response, *h(n)* of the weighted synthesis filter is computed.

- Closed-loop pitch analysis is then performed (to find the pitch lag and gain), using the target *x(n)* and impulse response *h(n)*, by searching around the open-loop pitch lag. Fractional pitch with 1/4th or 1/2nd of a sample resolution (depending on the mode and the pitch lag value) is used. The intrpolating filter in fractional pitch search has low pass frequency response. Further, there are two potential low-pass characteristics in the the adaptive codebook and this information is encoded with 1 bit.

- The target signal *x(n)* is updated by removing the adaptive codebook contribution (filtered adaptive codevector), and this new target, $x_2(n)$, is used in the fixed algebraic codebook search (to find the optimum innovation).

- The gains of the adaptive and fixed codebook are vector quantified with 6-7 bits (with moving average (MA) prediction applied to the fixed codebook gain).

- Finally, the filter memories are updated (using the determined excitation signal) for finding the target signal in the next subframe.

The bit allocation of the AMR-WB codec modes is shown in table 1. In each 20 ms speech frame, 132, 177, 253, 285, 317, 365, 397, 461 and 477 bits are produced, corresponding to a bit-rate of 6.60, 8.85 ,12.65, 14.25, 15.85, 18.25, 19.85, 23.05 or 23.85 kbit/s. More detailed bit allocation among the codec parameters is given in tables 12a-12i. Note that the most significant bits (MSB) are always sent first.

Table 1: Bit allocation of the AMR-WB coding algorithm for 20 ms frame

Mode	Parameter	1st subframe	2nd subframe	3rd subframe	4th subframe	total per frame
23.85 kbit/s	VAD-flag					1
	ISP					46
	LTP-filtering	1	1	1	1	4
	Pitch delay	9	6	9	6	30
	Algebraic code	88	88	88	88	352
	Codebook gain	7	7	7	7	28
	HB-energy	4	4	4	4	16
	Total					477
23.05 kbit/s	VAD-flag					1
	ISP					46
	LTP-filtering	1	1	1	1	4
	Pitch delay	9	6	9	6	30
	Algebraic code	88	88	88	88	352
	Gains	7	7	7	7	28
	Total					461
19.85 kbit/s	VAD-flag					1
	ISP					46
	LTP-filtering	1	1	1	1	4
	Pitch delay	9	6	9	6	30
	Algebraic code	72	72	72	72	288
	Codebook gain	7	7	7	7	28
	Total					397
18.25 kbit/s	VAD-flag					1
	ISP					46
	LTP-filtering	1	1	1	1	4
	Pitch delay	9	6	9	6	30
	Algebraic code	64	64	64	64	256
	Gains	7	7	7	7	28
	Total					365
15.85 kbit/s	VAD-flag					1
	ISP					46
	LTP-filtering	1	1	1	1	4
	Pitch delay	9	6	9	6	30
	Algebraic code	52	52	52	52	208
	Gains	7	7	7	7	28
	Total					317
14.25 kbit/s	VAD-flag					1
	ISP					46
	LTP-filtering	1	1	1	1	4
	Pitch delay	9	6	9	6	30
	Algebraic code	44	44	44	44	176
	Gains	7	7	7	7	28
	Total					285
12.65 kbit/s	VAD-flag					1
	ISP					46
	LTP-filtering	1	1	1	1	4
	Pitch delay	9	6	9	6	30
	Algebraic code	36	36	36	36	144
	Gains	7	7	7	7	28
	Total					253
8.85 kbit/s	VAD-flag					1
	ISP					46
	Pitch delay	8	5	8	5	26
	Algebraic code	20	20	20	20	80
	Gains	6	6	6	6	24
	Total					177
6.60 kbit/s	VAD-flag					1
	ISP					36
	Pitch delay	8	5	5	5	23
	Algebraic code	12	12	12	12	48
	Gains	6	6	6	6	24
	Total					132

4.4 Principles of the adaptive multi-rate speech decoder

The signal flow at the decoder is shown in Figure 3. At the decoder, the transmitted indices are extracted from the received bitstream. The indices are decoded to obtain the coder parameters at each transmission frame. These parameters are the ISP vector, the 4 fractional pitch lags, the 4 LTP filtering parameters, the 4 innovative codevectors, and the 4 sets of vector quantized pitch and innovative gains. In 23.85 kbit/s mode, also high-band gain index is decoded. The ISP vector is converted to the LP filter coefficients and interpolated to obtain LP filters at each subframe. Then, at each 64-sample subframe:

- The excitation is constructed by adding the adaptive and innovative codevectors scaled by their respective gains.
- The 12.8 kHz speech is reconstructed by filtering the excitation through the LP synthesis filter.
- The reconstructed speech is de-emphasized.

Finally, the reconstructed speech is upsampled to 16 kHz and high-band speech signal is added to the frequency band from 6 kHz to 7 kHz.

4.5 Sequence and subjective importance of encoded parameters

The encoder will produce the output information in a unique sequence and format, and the decoder must receive the same information in the same way. In table 12a-12i, the sequence of output bits and the bit allocation for each parameter is shown.

The different parameters of the encoded speech and their individual bits have unequal importance with respect to subjective quality. The output and input frame formats for the AMR wideband speech codec are given in [2], where a reordering of bits take place.

5 Functional description of the encoder

In this clause, the different functions of the encoder represented in Figure 2 are described.

5.1 Pre-processing

The encoder performs the analysis of the LPC, LTP and fixed codebook parameters at 12.8 kHz sampling rate. Therefore, the input signal has to be decimated from 16kHz to 12.8 kHz. The decimation is performed by first upsampling by 4, then filtering the output through lowpass FIR filter $H_{decim}(z)$ that has the cut off frequency at 6.4 kHz. Then, the signal is downsampled by 5. The filtering delay is compensated by adding zeroes into the end of the input vector.

After the decimation, two pre-processing functions are applied to the signal prior to the encoding process: high-pass filtering and pre-emphasizing (and signal down-scaling).

(Down-scaling consists of dividing the input by a factor of 2 to reduce the possibility of overflows in the fixed-point implementation.)

The high-pass filter serves as a precaution against undesired low frequency components. A filter at a cut off frequency of 50 Hz is used, and it is given by

$$H_{h1}(z) = \frac{0.989502 - 1.979004z^{-1} + 0.989502z^{-2}}{1 - 1.978882z^{-1} + 0.979126z^{-2}}. \quad (4)$$

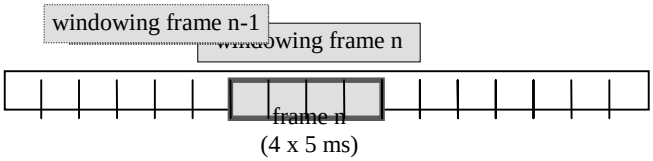
(Both down-scaling and high-pass filtering are combined by dividing the coefficients at the numerator of $H_{h1}(z)$ by 2.)

In the pre-emphasis, a first order high-pass filter is used to emphasize higher frequencies, and it is given by

$$H_{pre-emph}(z) = 1 - 0.68z^{-1} \quad (5)$$

5.2 Linear prediction analysis and quantization

Short-term prediction, or LP, analysis is performed once per speech frame using the autocorrelation approach with 30 ms asymmetric windows. An overhead of 5 ms is used in the autocorrelation computation. The frame structure is depicted below.



The autocorrelations of windowed speech are converted to the LP coefficients using the Levinson algorithm. Then the LP coefficients are transformed to the ISP domain for quantization and interpolation purposes. The interpolated quantized and unquantized filters are converted back to the LP filter coefficients (to construct the synthesis and weighting filters at each subframe).

5.2.1 Windowing and auto-correlation computation

LP analysis is performed once per frame using an asymmetric window. The window has its weight concentrated at the fourth subframe and it consists of two parts: the first part is a half of a Hamming window and the second part is a quarter of a Hamming-cosine function cycle. The window is given by:

$$\begin{aligned} w(n) &= 0.54 - 0.46\cos\left(\frac{2\pi n}{2L_1 - 1}\right), \quad n = 0,\ldots,L_1 - 1, \\ &= \cos\left(\frac{2\pi(n - L_1)}{4L_2 - 1}\right), \quad n = L_1,\ldots,L_1 + L_2 - 1 \end{aligned} \qquad (6)$$

where the values $L_1=256$ and $L_2=128$ are used.

The autocorrelations of the windowed speech $s'(n),n=0,\ldots,383$ are computed by

$$r(k) = \sum_{n=k}^{383} s'(n)s'(n-k), \quad k = 0,\ldots,16, \qquad (7)$$

and a 60 Hz bandwidth expansion is used by lag windowing the autocorrelations using the window [2]

$$w_{lag}(i) = \exp\left[-\frac{1}{2}\left(\frac{2\pi f_0 i}{f_s}\right)^2\right], \quad i = 1,\ldots 10, \qquad (8)$$

where $f_0=60$ Hz is the bandwidth expansion and $f_s=12800$ Hz is the sampling frequency. Further, $r(0)$ is multiplied by the white noise correction factor 1.0001 which is equivalent to adding a noise floor at -40 dB.

5.2.2 Levinson-Durbin algorithm

The modified autocorrelations $r'(0) = 1.0001\,r(0)$ and $r'(k) = r(k)w_{lag}(k), k = 1,\ldots 16,$ are used to obtain the LP filter coefficients $a_k,k=1,\ldots,16$ by solving the set of equations.

$$\sum_{k=1}^{16} a_k r'(|i-k|) = -r'(i), \quad i = 1,\ldots,16. \qquad (9)$$

The set of equations in (9) is solved using the Levinson-Durbin algorithm [3]. This algorithm uses the following recursion:

$$E(0) = r'(0)$$
For $i = 1$ to 16 do
$$k_i = -\left[r'(i) + \sum_{j=1}^{i-1} a_j^{i-1} r'(i-j)\right] / E(i-1)$$
$$a_i^{(i)} = k_i$$
For $j = 1$ to $i-1$ do
$$a_j^{(i)} = a_j^{(i-1)} + k_i a_{i-j}^{(i-1)}$$
$$E(i) = \left(1 - k_i^2\right) E(i-1)$$

The final solution is given as $a_j = a_j^{(16)}$,j=1,…,16.

5.2.3 LP to ISP conversion

The LP filter coefficients a_k,k=1,…,16, are converted to the ISP representation for quantization and interpolation purposes. For a 16th order LP filter, the ISPs are defined as the roots of the sum and difference polynomials

$$f_1'(z) = A(z) + z^{-16} A(z^{-1}) \quad (10)$$

and

$$f_2'(z) = A(z) - z^{-16} A(z^{-1}) \quad (11)$$

respectively. (The polynomials $f'_1(z)$ and $f'_2(z)$ are symmetric and antisymmetric, respectively). It can be proven that all roots of these polynomials are on the unit circle and they alternate each other [5]. $f'_2(z)$ has two roots at $z = 1$ (ω=0) and $z = -1$ ($\omega = \pi$). To eliminate these two roots, we define the new polynomials

$$f_1(z) = f_1'(z) \quad (12)$$

and

$$f_2(z) = f_2'(z)/(1 - z^{-2}). \quad (13)$$

Polynomials $f_1(z)$ and $f_2(z)$ have 8 and 7 conjugate roots on the unit circle $\left(e^{\pm j\omega_i}\right)$, respectively. Therefore, the polynomials can be written as

$$F_1(z) = (1 + a[16]) \prod_{i=0,2,\ldots,14} \left(1 - 2q_i z^{-1} + z^{-2}\right) \quad (14)$$

and

$$F_2(z) = (1 - a[16]) \prod_{i=1,3,\ldots,13} \left(1 - 2q_i z^{-1} + z^{-2}\right) \quad (15)$$

where $q_i = cos(\omega_i)$ with ω_i being the immittance spectral frequencies (ISF) and a[16] is the last predictor coefficient. ISFs satisfy the ordering property $0 < \omega_1 < \omega_2 < \ldots < \omega_{10} < \pi$. We refer to q_i as the ISPs in the cosine domain.

Since both polynomials $f_1(z)$ and $f_2(z)$ are symmetric only the first 8 and 7 coefficients of each polynomial, respectively, and the last predictor coefficient need to be computed.

The coefficients of these polynomials are found by the recursive relations (for i=0 to 7)

$$\begin{aligned} f_1(i+1) &= a_{i+1} + a_{m-i}, \\ f_2(i+1) &= a_{i+1} - a_{m-i} + f_2(i-1). \end{aligned} \quad (16)$$

where m=16 is the predictor order.

The ISPs are found by evaluating the polynomials $F_1(z)$ and $F_2(z)$ at 100 points equally spaced between 0 and π and checking for sign changes. A sign change signifies the existence of a root and the sign change interval is then divided 4 times to better track the root. The Chebyshev polynomials are used to evaluate $F_1(z)$ and $F_2(z)$ [6]. In this method the roots are found directly in the cosine domain $\{q_i\}$. The polynomials $F_1(z)$ and $F_2(z)$ evaluated at $z = e^{j\omega}$ can be written as

$$F(\omega) = 2e^{-j5\omega}C(x), \qquad (17)$$

with

$$C(x) = \sum_{i=0}^{7} f(i)T_{8-i}(x) + f(8)/2, \qquad (18)$$

where T_m=cos($m\omega$) is the m th order Chebyshev polynomial, and $f(i)$,i=1,…,8, are the coefficients of either $F_1(z)$ or $F_2(z)$, computed using the equations in (15). The polynomial $C(x)$ is evaluated at a certain value of x = cos(ω) using the recursive relation:

for $k = 7$ *down to* 1

$b_k = 2xb_{k+1} - b_{k+2} + f(8-k)$

end

$C(x) = xb_1 - b_2 + f(8)/2,$

with initial values b_8=1 and b_9=0. The details of the Chebyshev polynomial evaluation method are found in [6].

The LP filter coefficients are converted to the ISP representation [4] for quantization and interpolation purposes. The conversions to the ISP domain and back to the LP filter domain are described in the next section.

5.2.4 ISP to LP conversion

Once the ISPs are quantized and interpolated, they are converted back to the LP coefficient domain $\{a_k\}$. The conversion to the LP domain is done as follows. The coefficients of $F_1(z)$ and $F_2(z)$ are found by expanding Equations (14) and (15) knowing the quantized and interpolated ISPs q_i=,i=1,…,m, where m=16. The following recursive relation is used to compute $f_1(z)$

for $i = 2$ to $m/2$

$f_1(i) = -2q_{2i-2}f_1(i-1) + 2f_1(i-2)$

for $j = i-1$ down to 2

$f_1(j) = f_1(j) - 2q_{2i-2}f_1(j-1) + f_1(j-2)$

end

$f_1(1) = f_1(1) - 2q_{2i-2}$

end

with initial values $f_1(0)$=1 and $f_1(1)$=-2q_0. The coefficients $f_2(i)$ are computed similarly by replacing q_{2i-2} by q_{2i-1} and m/2 by m/2-1.

Once the coefficients $f_1(z)$ and $f_2(z)$ are found, $F_2(z)$ is multiplied by 1-z^{-2}, to obtain $F'_2(z)$; that is

$$\begin{aligned} f_2'(i) &= f_2(i) - f_2(i-2), \quad i = 2,\ldots,m/2-1, \\ f_1'(i) &= f_1(i) \end{aligned} \qquad (19)$$

Then $F_1(z)$ and $F'_2(z)$ are multiplied by 1+q_{m-1} and 1-q_{m-1}, respectively.

Finally the LP coefficients are found by

$$a_i = \begin{cases} 0.5 f_1'(i) + 0.5 f_2'(i), & i = 1,\ldots,m/2-1, \\ 0.5 f_1'(i) - 0.5 f_2'(i), & i = m/2+1,\ldots,m-1, \\ 0.5 f_1'(m/2)(1+q_{15}), & i = m/2, \\ q_{15}, & i = m. \end{cases} \quad (20)$$

This is directly derived from the relation $A(z) = (F_1'(z) + F_2'(z))/2$, and considering the fact that $F'_2(z)$ and $F'_2(z)$ are symmetric and antisymmetric polynomials, respectively.

5.2.5 Quantization of the ISP coefficients

The LP filter coefficients are quantized using the ISP representation in the frequency domain; that is

$$f_i = \begin{cases} \frac{f_S}{2\pi}\arccos(q_i), & i = 1,...,15, \\ \frac{f_S}{4\pi}\arccos(q_i), & i = 16 \end{cases} \quad (21)$$

where f_i are the ISFs in Hz [0,6400] and f_s=12800 is the sampling frequency. The ISF vector is given by $\mathbf{f}^t = [f_1 f_1,\ldots,f_1]$, with t denoting transpose.

A 1st order MA prediction is applied, and the residual ISF vector is quantified using a combination of split vector quantization (SVQ) and multi-stage vector quantization (MSVQ). The prediction and quantization are performed as follows. Let $\mathbf{z}(n)$ denote the mean-removed ISF vector at frame n. The prediction residual vector $\mathbf{r}(n)$ is given by:

$$\mathbf{r}(n) = \mathbf{z}(n) - \mathbf{p}(n) \quad (22)$$

where $\mathbf{p}(n)$ is the predicted LSF vector at frame n. First order moving-average (MA) prediction is used where:

$$\mathbf{p}(n) = \frac{1}{3}\hat{\mathbf{r}}(n-1), \quad (23)$$

where $\hat{\mathbf{r}}(n-1)$ is the quantified residual vector at the past frame.

The ISF residual vector $\mathbf{r}$ is quantized using split-multistage vector quantization S-MSVQ. The vector is split into 2 subvectors $\mathbf{r}_1(n)$ and $\mathbf{r}_2(n)$ of dimensions 9 and 7, respectively. The 2 subvectors are quantized in two stages. In the first stage $\mathbf{r}_1(n)$ is quantized with 8 bits and $\mathbf{r}_2(n)$ with 8 bits.

For 8.85 ,12.65, 14.25, 15.85, 18.25, 19.85, 23.05 or 23.85 kbit/s modes, the quantization error vectors $\mathbf{r}^{(2)}{}_i = \mathbf{r}_i - \mathbf{r}_i, i = 1,2$ are split in the next stage into 3 and 2 subvectors, respectively. The subvectors are quantized using the bit-rates described in Table 2.

Table 2. Quantization of ISP vector for the 8.85 ,12.65, 14.25, 15.85, 18.25, 19.85, 23.05 or 23.85 kbit/s modes

1. UNQUANTIZED 16-ELEMENT-LONG ISP VECTOR				
2. STAGE 1 ($\mathbf{r}_1$) 8 bits			**2. STAGE 1 ($\mathbf{r}_2$) 8 bits**	
3. STAGE 2 ($\mathbf{r}^{(2)}{}_{1,1-3}$) 6 bits	3. STAGE 2 ($\mathbf{r}^{(2)}{}_{1,4-6}$) 7 bits	3. STAGE 2 ($\mathbf{r}^{(2)}{}_{1,7-9}$) 7 bits	3. STAGE 2 ($\mathbf{r}^{(2)}{}_{2,1-3}$) 5 bits	3. STAGE 2 ($\mathbf{r}^{(2)}{}_{2,4-7}$) 5 bits

For 6.60 kbit/s mode, the quantization error vectors $\mathbf{r}^{(2)}{}_i = \mathbf{r}_i - \mathbf{r}_i, i = 1,2$ are split in the next stage into 2 and 1 subvectors, respectively. The subvectors are quantized using the bit-rates described in Table 3.

Table 3. Quantization of ISP vector for the 6.60 kbit/s mode

1. UNQUANTIZED 16-ELEMENT-LONG ISP VECTOR		
2. STAGE 1 ($\mathbf{r}_1$) 8 bits		2. STAGE 1 ($\mathbf{r}_2$) 8 bits
3. STAGE 2 ($\mathbf{r}^{(2)}_{1,1-5}$) 7 bits	3. STAGE 2 ($\mathbf{r}^{(2)}_{1,6-9}$) 7 bits	3. STAGE 2 ($\mathbf{r}^{(2)}_{2,1-7}$) 6 bits

A squared error ISP distortion measure is used in the quantization process. In general, for an input ISP or error residual subvector $\mathbf{r}_i$,i=1,2 and a quantized vector at index k, $\hat{\mathbf{r}}_i^k$, the quantization is performed by finding the index k which minimizes

$$E = \sum_{k=1}^{10} \left[f_i w_i - \hat{f}_i^k w_i \right]^2 , \qquad (24)$$

where m and n are the first and last elements of the subvector.

5.2.6 Interpolation of the ISPs

The set of quantized (and unquantized) LP parameters is used for the fourth subframe whereas the first, second, and third subframes use a linear interpolation of the parameters in the adjacent frames. The interpolation is performed on the ISPs in the $\mathbf{q}$ domain. Let $\hat{\mathbf{q}}_4^{(n)}$ be the ISP vector at the 4th subframe of the frame, and $\hat{\mathbf{q}}_4^{(n-1)}$ the ISP vector at the 4th subframe of the past frame n-1. The interpolated ISP vectors at the 1st, 2nd, and 3rd subframes are given by

$$\hat{\mathbf{q}}_1^{(n)} = 0.55\hat{\mathbf{q}}_4^{(n-1)} + 0.45\hat{\mathbf{q}}_4^{(n)},$$
$$\hat{\mathbf{q}}_2^{(n)} = 0.2\hat{\mathbf{q}}_4^{(n-1)} + 0.8\hat{\mathbf{q}}_4^{(n)},$$
$$\hat{\mathbf{q}}_3^{(n)} = 0.04\hat{\mathbf{q}}_4^{(n-1)} + 0.96\hat{\mathbf{q}}_4^{(n)}.$$

The same formula is used for interpolation of the unquantized ISPs. The interpolated ISP vectors are used to compute a different LP filter at each subframe (both quantized and unquantized) using the ISP to LP conversion method described in Section 5.2.4.

5.3 Perceptual weighting

The traditional perceptual weighting filter $W(z) = A(z/\gamma_1)/A(z/\gamma_2)$ has inherent limitations in modelling the formant structure and the required spectral tilt concurrently. The spectral tilt is more pronounced in wideband signals due to the wide dynamic range between low and high frequencies. A solution to this problem is to introduce the preemphasis filter at the input, compute the LP filter $A(z)$ based on the preemphasized speech $\mathbf{s}(n)$, and use a modified filter $W(z)$ by fixing its denominator. This structure substantially decouples the formant weighting from the tilt.

A weighting filter of the form $W(z) = A(z/\gamma_1)H_{de-emph}(z)$ is used, where $H_{de-emph} = \dfrac{1}{1-\beta_1 z^{-1}}$ and β_1=0.68.

Because $A(z)$ is computed based on the preemphasized speech signal $s(n)$, the tilt of the filter $1/A(z/\gamma_1)$ is less pronounced compared to the case when $A(z)$ is computed based on the original speech. Since deemphasis is performed at the decoder end, it can be shown that the quantization error spectrum is shaped by a filter having a transfer function $W^{-1}(z)H_{de\text{-}emph}(z)=1/A(z/\gamma_1)$. Thus, the spectrum of the quantization error is shaped by a filter whose transfer function is $1/A(z/\gamma_1)$, with $A(z)$ computed based on the preemphasized speech signal.

5.4 Open-loop pitch analysis

Open-loop pitch analysis is performed twice per frame (each 10 ms) to find two estimates of the pitch lag in each frame. This is done in order to simplify the pitch analysis and confine the closed loop pitch search to a small number of lags around the open-loop estimated lags.

Open-loop pitch estimation is based on the weighted speech signal $s_w(n)$ which is obtained by filtering the input speech signal through the weighting filter $W(z) = A(z/\gamma_1)H_{de-emph}(z)$, where $H_{de-emph} = \frac{1}{1-\beta_1 z^{-1}}$ and β_1=0.68. That is, in a subframe of size L, the weighted speech is given by

$$s_w(n) = s(n) + \sum_{i=1}^{16} a_i \gamma_1^i s(n-i) + \beta_1 s_w(n-1), n = 0,...,L-1. \quad (25)$$

The open-loop pitch analysis is performed to a signal by decimated two. The decimated signal is obtained by filtering $s_w(n)$ through a fourth order FIR filter $H_{decim2}(z)$ and then downsampling the output by two.

5.4.1 6.60 kbit/s mode

Open-loop pitch analysis is performed once per frame (every 20 ms) to find an estimate of the pitch lag in each frame.

The open-loop pitch analysis is performed as follows. First, the correlation of weighted speech is determined for each pitch lag value *d* by:

$$C(d) = \sum_{n=0}^{128} s_w(n) s_w(n-d) w(d), d = 17,\ldots,115, \quad (26)$$

where *w(d)* is a weighting function. The estimated pitch-lag is the delay that maximises the weighted correlation function *C(d)*. The weighting emphasises lower pitch lag values reducing the likelihood of selecting a multiple of the correct delay. The weighting function consists of two parts: a low pitch lag emphasis function, $w_l(d)$, and a previous frame lag neighbouring emphasis function, $w_n(d)$:

$$w(d) = w_l(d) w_n(d). \quad (27)$$

The low pitch lag emphasis function is a given by:

$$w_l(d) = cw(d) \quad (28)$$

where *cw(d)* is defined by a table in the fixed point computational description. The previous frame lag neighbouring emphasis function depends on the pitch lag of previous speech frames:

$$w_n(d) = \begin{cases} cw(|T_{old} - d| + 98), & v > 0.8 \text{ and } e < 12800, \\ 1.0, & \text{otherwise}, \end{cases} \quad (29)$$

where T_{old} is the median filtered pitch lag of 5 previous voiced speech half-frames, *v* is an adaptive parameter and e is the energy of the first half-frame. If the frame is classified as voiced by having the open-loop gain g>0.6 and the frame is not silence e<12800, then the v-value is set to 1.0 for the next frame. Otherwise, the v-value is updated by v=0.9v. The open loop gain is given by:

$$g = \frac{\sum_{n=0}^{128} s_w(n) s_w(n-d_{\max})}{\sqrt{\sum_{n=0}^{128} s_w^2(n) \sum_{n=0}^{128} s_w^2(n-d_{\max})}} \quad (30)$$

where d_{max} is the pitch delay that maximizes *C(d)*. The median filter is updated only during voiced speech frames. The weighting depends on the reliability of the old pitch lags. If previous frames have contained unvoiced speech or silence, the weighting is attenuated through the parameter *v*.

5.4.2 8.85, 12.65, 14.25, 15.85, 18.25, 19.85, 23.05 or 23.85 kbit/s modes

Open-loop pitch analysis is performed twice per frame (every 10 ms) to find two estimates of the pitch lag in each frame.

The open-loop pitch analysis is performed as follows. First, the correlation of weighted speech is determined for each pitch lag value *d* by:

$$C(d) = \sum_{n=0}^{64} s_w(n) s_w(n-d) w(d), d = 17,\ldots,115, \quad (31)$$

where *w(d)* is a weighting function. The estimated pitch-lag is the delay that maximises the weighted correlation function *C(d)*. The weighting emphasises lower pitch lag values reducing the likelihood of selecting a multiple of the correct delay. The weighting function consists of two parts: a low pitch lag emphasis function, $w_l(d)$, and a previous frame lag neighbouring emphasis function, $w_n(d)$:

$$w(d) = w_l(d) w_n(d). \quad (32)$$

The low pitch lag emphasis function is a given by:

$$w_l(d) = cw(d) \quad (33)$$

where *cw(d)* is defined by a table in the fixed point computational description. The previous frame lag neighbouring emphasis function depends on the pitch lag of previous speech frames:

$$w_n(d) = \begin{cases} cw(|T_{old} - d| + 98), & v > 0.8 \text{ and } e < 12800, \\ 1.0, & \text{otherwise}, \end{cases} \quad (34)$$

where T_{old} is the median filtered pitch lag of 5 previous voiced speech half-frames, *v* is an adaptive parameter and e is the energy of the analyzed half-frame. If the frame is classified as voiced by having the open-loop gain g>0.6 and the frame is not silence e<12800, then the v-value is set to 1.0 for the next frame. Otherwise, the v-value is updated by v=0.9v. The open loop gain is given by:

$$g = \frac{\sum_{n=0}^{64} s_w(n) s_w(n - d_{\max})}{\sqrt{\sum_{n=0}^{64} s_w^2(n) \sum_{n=0}^{64} s_w^2(n - d_{\max})}} \quad (35)$$

where d_{max} is the pitch delay that maximizes *C(d)*. The median filter is updated only during voiced speech frames. The weighting depends on the reliability of the old pitch lags. If previous frames have contained unvoiced speech or silence, the weighting is attenuated through the parameter *v*.

5.5 Impulse response computation

The impulse response, *h(n)*, of the weighted synthesis filter $H(z)W(z) = A(z/\gamma_1)H_{de-emph}(z)/\hat{A}(z)$ is computed each subframe. This impulse response is needed for the search of adaptive and fixed codebooks. The impulse response *h(n)* is computed by filtering the vector of coefficients of the filter $A(z/\gamma_1)$ extended by zeros through the two filters $1/\hat{A}(z)$ and $H_{de-emph}(z)$.

5.6 Target signal computation

The target signal for adaptive codebook search is usually computed by subtracting the zero-input response of the weighted synthesis filter $H(z)W(z) = A(z/\gamma_1)H_{de-emph}(z)/\hat{A}(z)$ from the weighted speech signal $s_w(n)$. This is performed on a subframe basis.

An equivalent procedure for computing the target signal, which is used in this codec, is the filtering of the LP residual signal *r(n)* through the combination of synthesis filter $1/\hat{A}(z)$ and the weighting filter $A(z/\gamma_1)H_{de-emph}(z)$. After determining the excitation for the subframe, the initial states of these filters are updated by filtering the difference between the LP residual and excitation. The memory update of these filters is explained in Section 5.9.

The residual signal *r(n)* which is needed for finding the target vector is also used in the adaptive codebook search to extend the past excitation buffer. This simplifies the adaptive codebook search procedure for delays less than the subframe size of 64 as will be explained in the next section. The LP residual is given by

$$r(n) = s(n) + \sum_{i=1}^{16} \hat{a}_i s(n-i), n = 0,...,64. \quad (36)$$

5.7 Adaptive codebook

Adaptive codebook search is performed on a subframe basis. It consists of performing closed loop pitch search, and then computing the adaptive codevector by interpolating the past excitation at the selected fractional pitch lag.

The adaptive codebook parameters (or pitch parameters) are the delay and gain of the pitch filter. In the adaptive codebook approach for implementing the pitch filter, the excitation is repeated for delays less than the subframe length. In the search stage, the excitation is extended by the LP residual to simplify the closed-loop search.

In 12.65, 14.25, 15.85, 18.25, 19.85, 23.05 or 23.85 kbit/s modes, in the first and third subframes, a fractional pitch delay is used with resolutions 1/4 in the range $\lfloor 34, 127\frac{3}{4} \rfloor$, resolutions 1/2 in the range $\lfloor 128, 159\frac{1}{2} \rfloor$, and integers only in the range [160, 231]. For the second and fourth subframes, a pitch resolution of 1/4 is always used in the range $\lfloor T_1\text{-}8, T_1+7\frac{3}{4} \rfloor$, where T_1 is nearest integer to the fractional pitch lag of the previous (1st or 3rd) subframe.

In 8.85 kbit/s mode, in the first and third subframes, a fractional pitch delay is used with resolutions 1/2 in the range $\lfloor 34, 91\frac{1}{2} \rfloor$, and integers only in the range [92, 231]. For the second and fourth subframes, a pitch resolution of 1/2 is always used in the range $\lfloor T_1\text{-}8, T_1+7\frac{1}{2} \rfloor$, where T_1 is nearest integer to the fractional pitch lag of the previous (1st or 3rd) subframe.

In 6.60 kbit/s mode, in the first subframe, a fractional pitch delay is used with resolutions 1/2 in the range $\lfloor 34, 91\frac{1}{2} \rfloor$, and integers only in the range [92, 231]. For the second, third and fourth subframes, a pitch resolution of 1/2 is always used in the range $\lfloor T_1\text{-}8, T_1+7\frac{1}{2} \rfloor$, where T_1 is nearest integer to the fractional pitch lag of the first subframe.

Closed-loop pitch analysis is performed around the open-loop pitch estimates on a subframe basis. In 8.85, 12.65, 14.25, 15.85, 18.25, 19.85, 23.05 or 23.85 kbit/s modes, in the first (and third) subframe the range T_{op}±7, bounded by 34...231, is searched. In 6.60 kbit/s mode, in the first subframe the range T_{op}±7, bounded by 34...231, is searched. For all the modes, for the other subframes, closed-loop pitch analysis is performed around the integer pitch selected in the previous subframe, as described above. In 12.65, 14.25, 15.85, 18.25, 19.85, 23.05 or 23.85 kbit/s modes, the pitch delay is encoded with 9 bits in the first and third subframes and the relative delay of the other subframes is encoded with 6 bits. In 8.85 kbit/s mode, the pitch delay is encoded with 8 bits in the first and third subframes and the relative delay of the other subframes is encoded with 5 bits. In 6.60 kbit/s mode, the pitch delay is encoded with 8 bits in the first subframe and the relative delay of the other subframes is encoded with 5 bits.

The closed loop pitch search is performed by minimizing the mean-square weighted error between the original and synthesized speech. This is achieved by maximizing the term

$$T_k = \frac{\sum_{n=0}^{63} x(n) y_k(n)}{\sqrt{\sum_{n=0}^{63} y_k(n) y_k(n)}}, \quad (37)$$

where $x(n)$ is the target signal and $y_k(n)$ is the past filtered excitation at delay k (past excitation convolved with $h(n)$). Note that the search range is limited around the open-loop pitch as explained earlier.

The convolution $y_k(n)$ is computed for the first delay in the searched range, and for the other delays, it is updated using the recursive relation

$$y_k(n) = y_{k-1}(n-1) + u(-k)h(n) \qquad (38)$$

where u(n),n=–(231+17),…,63, is the excitation buffer. Note that in search stage, the samples $u(n), n = 0,\ldots,63$, are not known, and they are needed for pitch delays less than 64. To simplify the search, the LP residual is copied to $u(n)$ in order to make the relation in Equation (38) valid for all delays.

Once the optimum integer pitch delay is determined, the fractions from $\frac{-3}{4}$ to $\frac{3}{4}$ with a step of $\frac{1}{4}$ around that integer are tested. The fractional pitch search is performed by interpolating the normalized correlation in Equation (37) and searching for its maximum. Once the fractional pitch lag is determined, $v'(n)$ is computed by interpolating the past excitation signal $u(n)$ at the given phase (fraction). (The interpolation is performed using two FIR filters (Hamming windowed sinc functions); one for interpolating the term in Equation (34) with the sinc truncated at ±17 and the other for interpolating the past excitation with the sinc truncated at ±63). The filters have their cut-off frequency (-3 dB) at 6000 Hz in the oversampled domain, which means that the interpolation filters exhibit low-pass frequency response Thus, even when the pitch delay is an integer value, the adaptive codebook excitation consists of a low-pass filtered version of the past excitation at the given delay and not a direct copy thereof.

Further, in order to enhance the pitch prediction performance in wideband signals, a frequency-dependant pitch predictor is used. This is important in wideband signals since the periodicity doesn't necessarily extend over the whole spectrum. In this algorithm, there are two signal paths associated to respective sets of pitch codebook parameters, wherein each signal path comprises a pitch prediction error calculating device for calculating a pitch prediction error of a pitch codevector from a pitch codebook search device. One of these two paths comprises a low-pass filter for filtering the pitch codevector and the pitch prediction error is calculated for these two signal paths. The signal path having the lowest calculated pitch prediction error is selected, along with the associated pitch gain.

The low pass filter used in the second path is in the form $B_{LP}(z)=0.18z+0.64+0.18z^{-1}$. Note that 1 bit is used to encode the chosen path.

Thus, for 8.85, 12.65, 14.25, 15.85, 18.25, 19.85, 23.05 or 23.85 kbit/s modes, there are two possibilities to generate the adaptive codebook $v(n)$, $v(n) = v'(n)$ in the first path, or $v(n) = \sum_{i=0}^{2} b_{LP}(i+1)v'(n-i)$ in the second path, where $\mathbf{b}_{LP}$=[0.18,0.64,0.18]. The path which results in minimum energy of the target signal $x_2(n)$ defined in Equation (37) is selected for the filtered adaptive codebook vector. For 6.60 kbit/s mode, $v(n)$ is always $v(n) = \sum_{i=0}^{2} b_{LP}(i+1)v'(n-i)$.

The adaptive codebook gain is then found by

$$g_p = \frac{\sum_{n=0}^{63} x(n)y(n)}{\sum_{n=0}^{63} y(n)y(n)}, \qquad \text{bounded by} \quad 0 \le g_p \le 1.2, \qquad (39)$$

where $y(n) = v(n) * h(n)$ is the filtered adaptive codebook vector (zero-state response of $H(z)W(z)$ to $v_i(n)$). To insure stability, the adaptive codebook gain g_p is bounded by 0.95, if the adaptive codebook gains of the previous subframes have been small and the LP filters of the previous subframes have been close to being unstable.

5.8 Algebraic codebook

5.8.1 Codebook structure

The codebook structure is based on interleaved single-pulse permutation (ISPP) design. The 64 positions in the codevector are divided into 4 tracks of interleaved positions, with 16 positions in each track. The different codebooks at

the different rates are constructed by placing a certain number of signed pulses in the tracks (from 1 to 6 pulses per track). The codebook index, or codeword, represents the pulse positions and signs in each track. Thus, no codebook storage is needed, since the excitation vector at the decoder can be constructed through the information contained in the index itself (no lookup tables).

An important feature of the used codebook is that it is a dynamic codebook consisting of an algebraic codebook followed by an adaptive prefilter $F(z)$ which enhances special spectral components in order to improve the synthesis speech quality. A prefilter relevant to wideband signals is used whereby $F(z)$ consists of two parts: a periodicity enhancement part $1/(1-0.85z^{-T})$ and a tilt part $(1 - \beta_1 z^{-1})$, where T is the integer part of the pitch lag and β_1 is related to the voicing of the previous subframe and is bounded by [0.0,0.5]. The codebook search is performed in the algebraic domain by combining the filter $F(z)$ with the weighed synthesis filter prior to the coddedbook search. Thus, the impulse response $h(n)$ must be modified to include the prefilter $F(z)$. That is, $h(n) \leftarrow h(n) * f(n)$.

The codebook structures of different bit rates are given below.

5.8.1.1 23.85 and 23.05 kbit/s mode

In this codebook, the innovation vector contains 24 non-zero pulses. All pulses can have the amplitudes +1 or -1. The 64 positions in a subframe are divided into 4 tracks, where each track contains six pulses, as shown in table 4.

Table 4. Potential positions of individual pulses in the algebraic codebook, 23.85 and 23.05 kbit/s.

Track	Pulse	Positions
1	$i_0, i_4, i_8, i_{12}, i_{16}, i_{20}$	0, 4, 8, 12, 16, 20, 24, 28, 32 36, 40, 44, 48, 52, 56, 60
2	$i_1, i_5, i_9, i_{13}, i_{17}, i_{21}$	1, 5, 9, 13, 17, 21, 25, 29, 33, 37, 41, 45, 49, 53, 57, 61
3	$i_2, i_6, i_{10}, i_{14}, i_{18}, i_{22}$	2, 6, 10, 14, 18, 22, 26, 30, 34, 38, 42, 46, 50, 54, 58, 62
4	$i_3, i_7, i_{11}, i_{15}, i_{19}, i_{23}$	3, 7, 11, 15, 19, 23, 27, 31, 35, 39, 43, 47, 51, 55, 59, 63

The six pulses in one track are encoded with 22 bits.

This gives a total of 88 bits (22+22+22+22) for the algebraic code.

5.8.1.2 19.85 kbit/s mode

In this codebook, the innovation vector contains 18 non-zero pulses. All pulses can have the amplitudes +1 or -1. The 64 positions in a subframe are divided into 4 tracks, where each track contains four pulses, as shown in table 5.

Table 5. Potential positions of individual pulses in the algebraic codebook, 19.85 kbit/s.

Track	Pulse	Positions
1	$i_0, i_4, i_8, i_{12}, i_{16}$	0, 4, 8, 12, 16, 20, 24, 28, 32 36, 40, 44, 48, 52, 56, 60
2	$i_1, i_5, i_9, i_{13}, i_{17}$	1, 5, 9, 13, 17, 21, 25, 29, 33, 37, 41, 45, 49, 53, 57, 61
3	i_2, i_6, i_{10}, i_{14}	2, 6, 10, 14, 18, 22, 26, 30, 34, 38, 42, 46, 50, 54, 58, 62
4	i_3, i_7, i_{11}, i_{15}	3, 7, 11, 15, 19, 23, 27, 31, 35, 39, 43, 47, 51, 55, 59, 63

The five pulses in one track are encoded with 20 bits. The four pulses in one track is encoded with 16 bits.

This gives a total of 72 bits (20+20+16+16) for the algebraic code.

5.8.1.3 18.25 kbit/s mode

In this codebook, the innovation vector contains 16 non-zero pulses. All pulses can have the amplitudes +1 or -1. The 64 positions in a subframe are divided into 4 tracks, where each track contains four pulses, as shown in table 6.

Table6. Potential positions of individual pulses in the algebraic codebook, 18.25 kbit/s.

Track	Pulse	Positions
1	i_0, i_4, i_8, i_{12}	0, 4, 8, 12, 16, 20, 24, 28, 32 36, 40, 44, 48, 52, 56, 60
2	i_1, i_5, i_9, i_{13}	1, 5, 9, 13, 17, 21, 25, 29, 33, 37, 41, 45, 49, 53, 57, 61
3	i_2, i_6, i_{10}, i_{14}	2, 6, 10, 14, 18, 22, 26, 30, 34, 38, 42, 46, 50, 54, 58, 62
4	i_3, i_7, i_{11}, i_{15}	3, 7, 11, 15, 19, 23, 27, 31, 35, 39, 43, 47, 51, 55, 59, 63

The four pulses in one track are encoded with 16 bits.

This gives a total of 64 bits (16+16+16+16) for the algebraic code.

5.8.1.4 15.85 kbit/s mode

In this codebook, the innovation vector contains 12 non-zero pulses. All pulses can have the amplitudes +1 or -1. The 64 positions in a subframe are divided into 4 tracks, where each track contains three pulses, as shown in table 7.

Table 7. Potential positions of individual pulses in the algebraic codebook, 15.85 kbit/s.

Track	Pulse	Positions
1	i_0, i_4, i_8	0, 4, 8, 12, 16, 20, 24, 28, 32 36, 40, 44, 48, 52, 56, 60
2	i_1, i_5, i_9	1, 5, 9, 13, 17, 21, 25, 29, 33, 37, 41, 45, 49, 53, 57, 61
3	i_2, i_6, i_{10}	2, 6, 10, 14, 18, 22, 26, 30, 34, 38, 42, 46, 50, 54, 58, 62
4	i_3, i_7, i_{11}	3, 7, 11, 15, 19, 23, 27, 31, 35, 39, 43, 47, 51, 55, 59, 63

The three pulses in one track are encoded with 13 bits.

This gives a total of 52 bits (13+13+13+13) for the algebraic code.

5.8.1.5 14.25 kbit/s mode

In this codebook, the innovation vector contains 10 non-zero pulses. All pulses can have the amplitudes +1 or -1. The 64 positions in a subframe are divided into 4 tracks, where each track contains two or three pulses, as shown in table 8.

Table 8. Potential positions of individual pulses in the algebraic codebook, 14.25 kbit/s.

Track	Pulse	Positions
1	i_0, i_4, i_8	0, 4, 8, 12, 16, 20, 24, 28, 32 36, 40, 44, 48, 52, 56, 60
2	i_1, i_5, i_9	1, 5, 9, 13, 17, 21, 25, 29, 33, 37, 41, 45, 49, 53, 57, 61
3	i_2, i_6	2, 6, 10, 14, 18, 22, 26, 30, 34, 38, 42, 46, 50, 54, 58, 62
4	i_3, i_7	3, 7, 11, 15, 19, 23, 27, 31, 35, 39, 43, 47, 51, 55, 59, 63

Each two pulse positions in one track are encoded with 8 bits (4 bits for the position of every pulse), and the sign of the first pulse in the track is encoded with 1 bit.

For two pulses located in the same track, only one sign bit is needed. This sign bit indicates the sign of the first pulse. The sign of the second pulse depends on its position relative to the first pulse. If the position of the second pulse is smaller, then it has opposite sign. Otherwise it has the same sign than in the first pulse.

The three pulse in one track are encoded with 13 bits.

This gives a total of 44 bits (13+13+9+9) for the algebraic code.

5.8.1.6 12.65 kbit/s mode

In this codebook, the innovation vector contains 8 non-zero pulses. All pulses can have the amplitudes +1 or -1. The 64 positions in a subframe are divided into 4 tracks, where each track contains two pulses, as shown in table 9.

Table 9. Potential positions of individual pulses in the algebraic codebook, 12.65 kbit/s.

Track	Pulse	Positions
1	i_0, i_4	0, 4, 8, 12, 16, 20, 24, 28, 32 36, 40, 44, 48, 52, 56, 60
2	i_1, i_5	1, 5, 9, 13, 17, 21, 25, 29, 33, 37, 41, 45, 49, 53, 57, 61
3	i_2, i_6	2, 6, 10, 14, 18, 22, 26, 30, 34, 38, 42, 46, 50, 54, 58, 62
4	i_3, i_7	3, 7, 11, 15, 19, 23, 27, 31, 35, 39, 43, 47, 51, 55, 59, 63

Each two pulse positions in one track are encoded with 8 bits (total of 32 bits, 4 bits for the position of every pulse), and the sign of the first pulse in the track is encoded with 1 bit (total of 4 bits).

For two pulses located in the same track, only one sign bit is needed. This sign bit indicates the sign of the first pulse. The sign of the second pulse depends on its position relative to the first pulse. If the position of the second pulse is smaller, then it has opposite sign. Otherwise it has the same sign than in the first pulse. This gives a total of 36 bits for the algebraic code.

5.8.1.7 8.85 kbit/s mode

In this codebook, the innovation vector contains 4 non-zero pulses. All pulses can have the amplitudes +1 or -1. The 64 positions in a subframe are divided into 4 tracks, where each track contains one pulse, as shown in table 10.

Table 10. Potential positions of individual pulses in the algebraic codebook, 8.85 kbit/s.

Track	Pulse	Positions
1	i_0	0, 4, 8, 12, 16, 20, 24, 28, 32 36, 40, 44, 48, 52, 56, 60
2	i_1	1, 5, 9, 13, 17, 21, 25, 29, 33, 37, 41, 45, 49, 53, 57, 61
3	i_2	2, 6, 10, 14, 18, 22, 26, 30, 34, 38, 42, 46, 50, 54, 58, 62
4	i_3	3, 7, 11, 15, 19, 23, 27, 31, 35, 39, 43, 47, 51, 55, 59, 63

Each pulse position in one track are encoded with 4 bits and the sign of the pulse in the track is encoded with 1 bit. This gives a total of 20 bits for the algebraic code.

5.8.1.8 6.60 kbit/s mode

In this codebook, the innovation vector contains 2 non-zero pulses. All pulses can have the amplitudes +1 or -1. The 64 positions in a subframe are divided into 2 tracks, where each track contains one pulse, as shown in table 11.

Table 11. Potential positions of individual pulses in the algebraic codebook, 6.60 kbit/s.

Track	Pulse	Positions
1	i_0	0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48, 50, 52, 54, 56, 58, 60, 62
2	i_1	1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 39, 41, 43, 45, 47, 49, 51, 53, 55, 57, 59, 61, 63

Each pulse position in one track are encoded with 5 bits and the sign of the pulse in the track is encoded with 1 bit. This gives a total of 12 bits for the algebraic code.

5.8.2 Pulse indexing

In the above section, the number of bits needed to encode a number of pulses in a track was given. In this section, the procedures used for encoding from 1 to 6 pulses per tack will be described. The description will be given for the case of 4 tracks per subframe, with 16 positions per track and pulse spacing of 4 (which is the case for all modes except the 6.6 kbit/s mode).

Encoding 1 signed pulse per track

The pulse position index is encoded with 4 bits and the sign index with 1 bit. The position index is given by the pulse position in the subframe divided by the pulse spacing (integer division). The division remainder gives the track index. For example, a pulse at position 31 has a position index of 31/4 = 7 and it belong to the track with index 3 (4^{th} track).

The sign index here is set to 0 for positive signs and 1 for negative signs.

The index of the signed pulse is given by

$$I_{1p} = p + s \times 2^{M}$$

where p is the position index, s is the sign index, and M=4 is the number of bits per tracks.

Encoding 2 signed pulses per track

In case of two pulses per track of $K=2^M$ potential positions (here M=4), each pulse needs 1 bit for the sign and M bits for the position, which gives a total of $2M+2$ bits. However, some redundancy exists due to the unimportance of the pulse ordering. For example, placing the first pulse at position p and the second pulse at position q is equivalent to placing the first pulse at position q and the second pulse at position p. One bit can be saved by encoding only one sign and deducing the second sign from the ordering of the positions in the index. Here the index is given by

$$I_{2p} = p_1 + p_0 \times 2^{M} + s \times 2^{2M}$$

where s is the sign index of the pulse at position index p_0. If the two signs are equal then the smaller position is set to p_0 and the larger position is set to p_1. On the other hand, of the two signs are not equal then the larger position is set to p_0 and the smaller position is set to p_1. At the decoder, the sign of the pulse at position p_0 is readily available. The second sign is deduced for the pulse ordering. If p_1 is larger than p_0 then the sign of the pulse at position p_1 is opposite to that at position p_0. If this is not the case then the two signs are set equal

Encoding 3 signed pulses per track

In case of three pulses per track, similar logic can be used as the case of two pulses. For a track with 2^M positions, 3M+1 bits are needed instead of 3M+3 bits. A simple way of indexing the pulses is to divide the track positions in two sections (or halves) and identify a section that contains at least two pulses. The number of positions in the section is $K/2 = 2^M/2 = 2^{M-1}$, which can be represented with M-1 bits. The two pulses in the section containing at least two pulses are encoded with the procedure for encoding 2 signed pulses which requires 2(M-1)+1 bits and the remaining pulse which can be anywhere in the track (in either section) is encoded with the M+1 bits. Finally, the index of the section that contains the two pulses is encoded with 1 bit. Thus the total number of required bits is 2(M-1)+1 + M+1 + 1 = 3M+1.

A simple way of checking if two pulses are positioned in the same section is done by checking whether their most significant bits (MSB) of their position indices are equal or not. Note that a MSB of 0 means that the position belongs to the lower half of the track (0-7) and MSB of 1 means it belongs to the upper half (8-15). If the two pulses belong to the upper half, they need to be shifted to the range (0-7) before encoding them using 2×3+1 bits. This can be done by masking the M-1 least significant bits (LSB) with a mask consisting of M-1 ones (which corresponds to the number 7 in this case).

The index of the 3 signed pulses is given by

$$I_{3p} = I_{2p} + k \times 2^{2M-1} + I_{1p} \times 2^{2M}$$

where I_{2p} is the index of the two pulses in the same section, k is the section index (0 or 1), and I_{1p} is the index of the third pulse in the track.

Encoding 4 signed pulses per track

The 4 signed pulses in a track of length $K=2^M$ can be encoded using $4M$ bits. Similar to the case of 3 pulses, the K positions in the track are divided into 2 sections (two halves) where each section contains $K/2=8$ positions. Here we denote the sections as Section A with positions 0 to $K/2$-1 and Section B with positions $K/2$ to K-1. Each section can

contain from 0 to 4 pulses. The table below shows the 5 cases representing the possible number of pulses in each sections

The table below shows the 5 cases representing the possible number of pulses in each sections:

case	*Pulses in Section A*	*Pulses in Section B*	*Bits needed*
0	0	4	4*M*-3
1	1	3	4*M*-2
2	2	2	4*M*-2
3	3	1	4*M*-2
4	4	0	4*M*-3

In cases 0 or 4, the 4 pulses in a section of length $K/2=2^{M-1}$ can be encoded using $4(M-1)+1=4M-3$ bits (this will be explained later on).

In cases 1 or 3, the 1 pulse in a section of length $K/2=2^{M-1}$ can be encoded with $M-1+1 = M$ bits and the 3 pulses in the other section can be encoded with $3(M-1)+1 = 3M-2$ bits. This gives a total of $M+3M-2 = 4M-2$ bits.

In case 2, the pulses in a section of length $K/2=2^{M-1}$ can be encoded with $2(M-1)+1 = 2M-1$ bits. Thus for both sections, $2(2M-1) = 4M-2$ bits are required.

Now the case index can be encoded with 2 bits (4 possible cases) assuming cases 0 and 4 are combined. Then for cases 1, 2, or 3, the number of needed bits is $4M-2$. This gives a total of $4M-2 + 2 = 4M$ bits. For cases 0 or 4, one bit is needed for identifying either case, and $4M-3$ bits are needed for encoding the 4 pulses in the section. Adding the 2 bits needed for the general case, this gives a total of $1+4M-3+2= 4M$ bits.

The index of the 4 signed pulses is given by

$$I_{4p} = I_{AB} + k\times 2^{4M-2}$$

where k is the case index (2 bits), and I_{AB} is the index of the pulses in both sections for each individual case.

For cases 0 and 1, I_{AB} is given by

$$I_{AB\_0,4} = I_{4p\_section} + j\times 2^{4M-3}$$

where j is a 1-bit index identifying the section with 4 pulses and $I_{4p\_section}$ is the index of the 4 pulses in that section (which requires 4M-3 bits).

For case 1, I_{AB} is given by

$$I_{AB\_1} = I_{3p\_B} + I_{1p\_A} \times 2^{3(M-1)+1}$$

where I_{3p_B} is the index of the 3 pulses in Section B (3(M-1)+1 bits) and I_{1p_A} is the index of the pulse in Section A ((M-1)+1 bits).

For case 2, I_{AB} is given by

$$I_{AB\_2} = I_{2p\_B} + I_{2p\_A} \times 2^{2(M-1)+1}$$

where I_{2p_B} is the index of the 2 pulses in Section B (2(M-1)+1 bits) and I_{2p_A} is the index of the two pulses in Section A (2(M-1)+1 bits).

Finally, for case 3, I_{AB} is given by

$$I_{AB\_3} = I_{1p\_B} + I_{3p\_A} \times 2^{M}$$

where I_{1p_B} is the index of the pulse in Section B ((M-1)+1 bits) and I_{3p_A} is the index of the 3 pulses in Section A (3(M-1)+1 bits).

For cases 0 and 4, it was mentioned that the 4 pulses in one section are encoded using $4(M-1)+1$ bits. This is done by further dividing the section into 2 subsections of length $K/4=2^{M-2}$ (=4 in this case); identifying a subsection that contains at least 2 pulses; coding the 2 pulses in that subsection using $2(M-2)+1=2M-3$ bits; coding the index of the subsection

that contains at least 2 pulses using 1 bit; and coding the remaining 2 pulses, assuming that they can be anywhere in the section, using 2(*M*-1)+1=2*M*-1 bits. This gives a total of (2M-3)+(1)+(2M-1) = 4M-3 bits

Encoding 5 signed pulses per track

The 5 signed pulses in a track of length $K=2^M$ can be encoded using 5*M* bits. Similar to the case of 4 pulses, the *K* positions in the track are divided into 2 sections A and B. Each section can contain from 0 to 5 pulses. A simple approach to encode the 5 pulses is to identify a section that contains at least 3 pulses and to encode the 3 pulses in that section using 3(*M*-1)+1= 3*M*-2 bits, and to encode the remaining 2 pulses in the whole track using 2*M*+1 bits. This gives 5*M*-1 bits. An extra bit is needed to identify the section that contains at least 3 pulses. Thus a total of 5*M* bits are needed to encode the 5 signed pulses.

The index of the 5 signed pulses is given by

$$I_{5p} = I_{2p} + I_{3p} \times 2^{2M} + k \times 2^{5M-1}$$

Where k is the index of the section that contains at least 3 pulses, I_{3p} is the index of the 3 pulses in that section (3(M-1)+1 bits), and I_{2p} is the index of the remaining 2 pulses in the track (2M+1 bits).

Encoding 6 signed pulses per track

The 6 signed pulses in a track of length $K=2^M$ are encoded using 6*M*-2 bits. Similar to the case of 5 pulses, the *K* positions in the track are divided into 2 sections A and B. Each section can contain from 0 to 6 pulses. The table below shows the 7 cases representing the possible number of pulses in each sections:

case	*Pulses in Section A*	*Pulses in Section B*	*Bits needed*
0	0	6	6*M*-5
1	1	5	6*M*-5
2	2	4	6*M*-5
3	3	3	6*M*-4
4	4	2	6*M*-5
5	5	1	6*M*-5
6	6	0	6*M*-5

Note that cases 0 and 6 are similar except that the 6 pulses are in different section. Similarly, cases 1 and 5 as well as cases 2 and 4 differ only in the section that contains more pulses. Therefore these cases can be coupled and an extra bit can be assigned to identify the section that contains more pulses. Since these cases initially need 6*M*-5 bits, the coupled cases need 6*M*-4 bits taking into account the Section bit. Thus, we have now 4 states of coupled cases, that is (0,6), (1,5), (2,4), and (3),with 2 extra bits needed for the state. This gives a total of 6*M*-4+2=6*M*-2 bits for the 6 signed pulses.

In cases 0 and 6, 1 bit is needed to identify the section which contains 6 pulses. 5 pulses in that section are encoded using 5(*M*-1) bits (since the pulses are confined to that section), and the remaining pulse is encoded using (*M*-1)+1 bits. Thus a total of 1+5(*M*-1)+*M*=6*M*-4 bits are needed for this coupled case. Extra 2 bits are needed to encode the state of the coupled case, giving a total of 6*M*-2 bits. For this coupled case, the index of the 6 pulses is given by

$$I_{6p} = I_{1p} + I_{5p} \times 2^{M} + j \times 2^{6M-5} + k \times 2^{6M-4}$$

where k is the index of the coupled case (2 bits), j is the index of the section containing 6 pulses (1 bit), I_{5p} is the index of 5 pulses in that section (5(M-1) bits), and I_{1p} is the index of the remaining pulse in that section ((M-1)+1 bits).

In cases 1 and 5, 1 bit is needed to identify the section which contains 5 pulses. The 5 pulses in that section are encoded using 5(*M*-1) bits and the pulse in the other section is encoded using (*M*-1)+1 bits. For this coupled case, the index of the 6 pulses is given by

$$I_{6p} = I_{1p} + I_{5p} \times 2^{M} + j \times 2^{6M-5} + k \times 2^{6M-4}$$

where k is the index of the coupled case (2 bits), j is the index of the section containing 5 pulses (1 bit), I_{5p} is the index of 5 pulses in that section (5(M-1) bits), and I_{1p} is the index of the pulse in the other section ((M-1)+1 bits).

In cases 2 or 4, 1 bit is needed to identify the section which contains 4 pulses. The 4 pulses in that section are encoded using 4(*M*-1) bits and the 2 pulses in the other section are encoded using 2(*M*-1)+1 bits. For this coupled case, the index of the 6 pulses is given by

$I_{6p} = I_{2p} + I_{4p} \times 2^{2(M-1)+1} + j \times 2^{6M-5} + k \times 2^{6M-4}$

where k is the index of the coupled case (2 bits), j is the index of the section containing 4 pulses (1 bit), I_{4p} is the index of 4 pulses in that section (4(M-1) bits), and I_{2p} is the index of the 2 pulses in the other section (2(M-1)+1 bits).

In case 3, the 3 pulses in each section are encoded using 3(*M*-1)+1 bits in each Section. For this case, the index of the 6 pulses is given by

$I_{6p} = I_{3pB} + I_{3pA} \times 2^{3(M-1)+1} + k \times 2^{6M-4}$

where k is the index of the coupled case (2 bits), I_{3pB} is the index of 3 pulses Section B (3(M-1)+1 bits), and I_{3pA} is the index of the 3 pulses in Section A (3(M-1)+1 bits).

5.8.3 Codebook search

The algebraic codebook is searched by minimizing the mean square error between the weighted input speech and the weighted synthesis speech. The target signal used in the closed-loop pitch search is updated by subtracting the adaptive codebook contribution. That is

$$x_2(n) = x(n) - g_p y(n), \quad n = 0,\ldots,63, \qquad (40)$$

where $y(n) = v(n) * h(n)$ is the filtered adaptive codebook vector and g_p is the unquantized adaptive codebook gain.

The matrix $\mathbf{H}$ is defined as the lower triangular Toeplitz convolution matrix with diagonal $h(0)$ and lower diagonals $h(1),\ldots,h(63)$, and $\mathbf{d} = \mathbf{H}^t\mathbf{x}_2$ is the correlation between the target signal $x_2(n)$ and the impulse response $h(n)$ (also known as the backward filtered target vector), and $\Phi = \mathbf{H}^t\mathbf{H}$ is the matrix of correlations of $h(n)$.

The elements of the vector **d** are computed by

$$d(n) = \sum_{i=n}^{63} x_2(i)h(i-n), \quad n = 0,\ldots 63, \qquad (41)$$

and the elements of the symmetric matrix Φ are computed by

$$\phi(i,j) = \sum_{n=j}^{63} h(n-i)h(n-j), \quad i = 0,...,63, \quad j = i,...,63. \qquad (42)$$

If $\mathbf{c}_k$ is the algebraic codevector at index *k*, then the algebraic codebook is searched by maximizing the search criterion

$$Q_k = \frac{(\mathbf{x}_2^t\mathbf{H}\mathbf{c}_k)^2}{\mathbf{c}_k^t\mathbf{H}^t\mathbf{H}c_k} = \frac{(\mathbf{d}^t\mathbf{c}_k)^2}{\mathbf{c}_k^t\Phi c_k} = \frac{(R_k)^2}{E_k}. \qquad (43)$$

The vector **d** and the matrix Φ are usually computed prior to the codebook search.

The algebraic structure of the codebooks allows for very fast search procedures since the innovation vector $\mathbf{c}_k$ contains only a few nonzero pulses. The correlation in the numerator of Equation (43) is given by

$$C = \sum_{i=0}^{N_p-1} a_i d(m_i) \qquad (44)$$

where m_i is the position of the ith pulse, a_i is its amplitude, and N_p is the number of pulses. The energy in the denominator of Equation (43) is given by

$$\mathrm{E} = \sum_{i=0}^{N_p-1} \phi(m_i, m_i) + 2 \sum_{i=0}^{N_p-2} \sum_{j=i+1}^{N_p-1} a_i a_j \phi(m_i, m_j) \qquad (45)$$

To simplify the search procedure, the pulse amplitudes are predetermined based on a certain reference signal $b(n)$. In this so-called signal-selected pulse amplitude approach, the sign of a pulse at position i is set equal to the sign of the reference signal at that position. Here, the reference signal $b(n)$ is given by

$$b(n) = \sqrt{\frac{E_d}{E_r}} r_{LTP}(n) + \alpha d(n) \qquad (46)$$

where $E_d = \mathbf{d}^t \mathbf{d}$ is the energy of the signal $d(n)$ and $E_r = \mathbf{r}_{LTP}^t \mathbf{r}_{LTP}$ is the energy of the signal $r_{LTP}(n)$ which is the residual signal after long term prediction. The scaling factor α controls the amount of dependence of the reference signal on $d(n)$, and it is lowered as the bit rate is increased. Here α=2 for 6.6 and 8.85 modes; α=1 for 12.65, 14.25, and 15.85 modes; α=0.8 for 18.25 mode; α=0.75 fo r19.85 mode; and α=0.5 for 23.05 and 23.85 modes.

To simplify the search the signal $d(n)$ and matrix Φ are modified to incorporate the pre-selected signs. Let s_b(n) denote the vector containing the signs of $b(n)$. The modified signal $d'(n)$ is given by

$$d'(n) = s_b(n) d(n) \qquad n=0,\ldots,N\text{-}1$$

and the modified autocorrelation matrix Φ' is given by

$$\phi'(i, j) = s_b(i) s_b(j) \phi(i, j), \qquad i=0,\ldots,N\text{-}1; \quad j=i,\ldots,N\text{-}1.$$

The correlation at the numerator of the search criterion Q_k is now given by

$$R = \sum_{i=0}^{N_p-1} d'(i)$$

and the energy at the denominator of the search criterion Q_k is given by

$$E = \sum_{i=0}^{N_p-1} \phi'(m_i, m_i) + 2 \sum_{i=0}^{N_p-2} \sum_{j=i+1}^{N_p-1} \phi'(m_i, m_j)$$

The goal of the search now is to determine the codevector with the best set of N_p pulse positions assuming amplitudes of the pulses have been selected as described above. The basic selection criterion is the maximization of the above mentioned ratio Q_k.

In order to reduce the search complexity, a fast search procedure known as depth-first tree search procedure is used, whereby the pulse positions are determined N_m pulses at a time. More precisely, the N_p available pulses are partitioned into M non-empty subsets of N_m pulses respectively such that $N_1+N_2...+N_m...+N_M = N_p$. A particular choice of positions for the first $J = N_1+N_2...+N_{m-1}$ pulses considered is called a level-m path or a path of length J. The basic criterion for a path of J pulse positions is the ratio $Q_k(J)$ when only the J relevant pulses are considered.

The search begins with subset #1 and proceeds with subsequent subsets according to a tree structure whereby subset m is searched at the m^{th} level of the tree. The purpose of the search at level 1 is to consider the N_1 pulses of subset #1 and their valid positions in order to determine one, or a number of, candidate path(s) of length N_1 which are the tree nodes at level l. The path at each terminating node of level m-1 is extended to length $N_1+N_2...+N_m$ at level m by considering N_m new pulses and their valid positions. One, or a number of, candidate extended path(s) are determined to constitute level-m nodes. The best codevector corresponds to that path of length N_p which maximizes the criterion $Q_k(N_p)$ with respect to all level-M nodes.

A special form of the depth-first tree search procedure is used here, in which two pulses are searched at a time, that is, N_m=2, and these 2 pulses belong to two consecutive tracks. Further, instead of assuming that the matrix Φ is

precomputed and stored, which requires a memory of $N \times N$ words (64×64= 4k words), a memory-efficient approach is used which reduces the memory requirement. In this approach, the search procedure is performed in such a way that only a part of the needed elements of the correlation matrix are precomputed and stored. This part corresponds to the correlations of the impulse response corresponding to potential pulse positions in consecutive tracks, as well as the correlations corresponding to $\phi(j,j)$, j=0,…,N-1 (that is the elements of the main diagonal of matrix Φ).

In order to reduce complexity, while testing possible combinations of two pulses, a limited number of potential positions of the first pulse are tested. Further, in case of large number of pulses, some pulses in the higher levels of the search tree are fixed. In order to guess intelligently which potential pulse positions are considered for the first pulse or in order to fix some pulse positions, a "pulse-position likelihood-estimate vector" **b** is used, which is based on speech-related signals. The p^{th} component $b(p)$ of this estimate vector **b** characterizes the probability of a pulse occupying position p (p = 0, 1, ... N-1) in the best codevector we are searching for. Here the estimate vector **b** is the same vector used for preselecting the amplitudes and given in Equation (46).

The search procedures for all bit rate modes are similar. Two pulses are searched at a time, and these two pulses always correspond to consecutive tracks. That is the two searched pulses are in tracks T_0-T_1, T_1-T_2, T_2-T_3, or T_3-T_0.

Before searching the positions, the sign of a pulse a potential position n is set the sign of $b(n)$ at that position. Then the modified signal $d'(n)$ is computed as described above by including the predetermined signs.

For the first 2 pulses (1st tree level), the correlation at the numerator of the search criterion is given by

$$R = d'(m_0) + d'(m_1)$$

and the energy at the denominator of the search criterion Q_k is given by

$$E = \phi'(m_0, m_0) + \phi'(m_1, m_1) + 2\phi'(m_0, m_1)$$

where the correlations $\phi'(m_i, m_j)$ has been modified to include the preselected signs at positions m_i and m_j.

For subsequent levels, the numerator and denominator are updated by adding the contribution of two new pulses. Assuming that two new pulses at a certain tree level with positions m_k and m_{k+1} from two consecutive tracks are searched, then the updated value of R is given by

$$R = R + d'(m_k) + d'(m_{k+1}) \tag{47}$$

and the updated energy is given by

$$E = E + \phi'(m_k, m_k) + \phi'(m_{k+1}, m_{k+1}) + 2\phi'(m_k, m_{k+1}) + 2R_{hv}(m_k) + 2R_{hv}(m_{k+1}) \tag{48}$$

where $R_{hv}(m)$ is the correlation between the impulse response $h(n)$ and a vector $v_h(n)$ containing the addition of delayed versions of impulse response at the previously determined positions. That is,

$$v_h(n) = \sum_{i=0}^{k-1} h(n - m_i)$$

and

$$R_{hv}(m) = \sum_{n=m}^{N-1} h(n) v_h(n - m)$$

At each tree level, the values of $R_{hv}(m)$ are computed online for all possible positions in each of the two tracks being tested. It can be seen from Equation (48) that only the correlations $\phi'(m_k, m_{k+1})$ corresponding to pulse positions in two consecutive tracks need to be stored (4×16×16 words), along with the correlations $\phi'(m_k, m_k)$ corresponding to the diagonal of the matrix Φ (64 words). Thus the memory requirement in the present algebraic structure is 1088 words instead of 64×64=4096 words.

The search procedures at the different bit rates modes are similar. The difference is in the number of pulses, and accordingly, the number of levels in the tree search. In order to keep a comparable search complexity across the different codebooks, the number of tested positions is kept similar.

The search in the 12.65 kbit/s mode will be described as an example. In this mode, 2 pulses are placed in each track giving a total of 8 pulses per subframe of length 64. Two pulses are searched at a time, and these two pulses always correspond to consecutive tracks. That is the two searched pulses are in tracks T_0-T_1, T_1-T_2, T_2-T_3, or T_3-T_0. The tree has 4 levels in this case. At the first level, pulse P_0 is assigned to track T_0 and pulse P_1 to track T_1. In this level, no search is performed and the two pulse positions are set to the maximum of $b(n)$ in each track. In the second level, pulse P_2 is assigned to track T_2 and pulse P_3 to track T_3. 4 positions for pulse P_2 are tested against all 16 positions of pulse P_3. The 4 tested positions of $P_2$are determined based on the maxima of $b(n)$ in the track. In the third level, pulse P_4 is assigned to track T_1 and pulse P_5 to track T_2. 8 positions for pulse P_4 are tested against all 16 positions of pulse P_5. Similar to the previous search level, the 8 tested positions of P_4 are determined based on the maxima of $b(n)$ in the track. In the fourth level, pulse P_6 is assigned to track T_3 and pulse P_7 to track T_0. 8 positions for pulse P_6 are tested against all 16 positions of pulse P_7. Thus the total number of tested combination is 4×16+8×16+8×16=320. The whole process is repeated 4 times (4 iterations) by assigning the pulses to different tracks. For example, in the 2nd iteration, pulses P_0 to P_7 are assigned to tracks T_1, T_2, T_3, T_0, T_2, T_3, T_0, and T_1, respectively. Thus the total number of tested position combinations is 4×320=1280.

As another search example, in the 15.85 kbit/s mode, 3 pulses are placed in each track giving a total of 12 pulses. There are 6 levels in the tree search whereby two pulses are searched in each level. In the first two levels, 4 pulses are set to the maxima of $b(n)$. In the subsequent 4 levels, the number of tested combinations are 4×16, 6×16, 8×16, and 8×16, respectively. 4 iterations are used giving a total of 4×26×16=1664 combinations.

5.9 Quantization of the adaptive and fixed codebook gains

The adaptive codebook gain (pitch gain) and the fixed (algebraic) codebook gain are vector quantized using a 6-bit codebook for modes 8.85 and 6.60 kbit/s and using a 7-bit codebook for all the other modes.

The fixed codebook gain quantization is performed using MA prediction with fixed coefficients. The 4th order MA prediction is performed on the innovation energy as follows. Let $E(n)$ be the mean-removed innovation energy (in dB) at subframe n, and given by

$$E(n) = 10\log\left(\frac{1}{N}g_c^2\sum_{i=0}^{N-1} c^2(i)\right) - \overline{E} \qquad (49)$$

where N=64 is the subframe size, $c(i)$ is the fixed codebook excitation, and $\overline{E} = 30$ dB is the mean of the innovation energy. The predicted energy is given by

$$\tilde{E}(n) = \sum_{i=1}^{4} b_i \hat{R}(n-i) \qquad (50)$$

where $[b_1 b_2 b_3]$=[05.,0.4,0.3,0.2] are the MA prediction coefficients, and $\hat{R}(k)$ is the quantized energy prediction error at subframe k. The predicted energy is used to compute a predicted fixed-codebook gain g'_c as in Equation (49) (by substituting $E(n)$ by $\tilde{E}(n)$ and g_c by g'_c). This is done as follows. First, the mean innovation energy is found by

$$E_i = 10\log\left(\frac{1}{N}\sum_{i=0}^{N-1} c^2(i)\right) \qquad (51)$$

and then the predicted gain g'_c is found by

$$g'_c = 10^{0.05(\tilde{E}(n)+\overline{E}-E_i)}. \qquad (52)$$

A correction factor between the gain g_c and the estimated one g'_c is given by

$$\gamma = g_c / g'_c. \qquad (53)$$

Note that the prediction error is given by

$$R(n) = E(n) - \tilde{E}(n) = 20 \log (\gamma). \quad (54)$$

The pitch gain, g_p, and correction factor γ are jointly vector quantized using a 6-bit codebook for modes 8.85 and 6.60 kbit/s, and 7-bit codebook for other modes. The gain codebook search is performed by minimizing the mean-square of the weighted error between original and reconstructed speech which is given

$$E = x^t x + g_p^2 y^t y + g_c^2 z^t z - 2g_p x^t y - 2g_c x^t z + 2g_p g_c y^t z, \quad (55)$$

where the x is the target vector, y is the filtered adaptive codebook vector, and z is the filtered fixed codebook vector. (Each gain vector in the codebook also has an element representing the quantized energy prediction error.) The quantized energy prediction error associated with the chosen gains is used to update $\hat{R}(n)$. In the search, only the 64 codevectors that are closest to the unquantized pitch gain, g_p, are taken into account.

5.10 Memory update

An update of the states of the synthesis and weighting filters is needed in order to compute the target signal in the next subframe.

After the two gains have been quantized, the excitation signal, $u(n)$, in the present subframe is found by

$$u(n) = \hat{g}_p v(n) + \hat{g}_c c(n), \quad n = 0,\ldots,63, \quad (56)$$

where $\hat{g}_p$ and $\hat{g}_c$ are the quantized adaptive and fixed codebook gains, respectively, $v_i(n)$ the adaptive codebook vector (interpolated past excitation), and $c(n)$ is the fixed codebook vector (algebraic code including pitch sharpening). The states of the filters can be updated by filtering the signal $r(n)–u(n)$ (difference between residual and excitation) through the filters $1/\hat{A}(z)$ and $A(z/\gamma_1)H_{de-emph}(z)$ for the 64 sample subframe and saving the states of the filters. This would require 3 filterings. A simpler approach which requires only one filtering is as follows. The local synthesis speech, $\hat{s}(n)$, is computed by filtering the excitation signal through $1/\hat{A}(z)$. The output of the filter due to the input $r(n)–u(n)$ is equivalent to $e(n) = s(n) - \hat{s}(n)$. So the states of the synthesis filter $1/\hat{A}(z)$ are given by $e(n)$,n=48,…,63. Updating the states of the filter $A(z/\gamma_1)H_{de-emph}(z)$ can be done by filtering the error signal $e(n)$ through this filter to find the perceptually weighted error $e_w(n)$. However, the signal $e_w(n)$ can be equivalently found by

$$e_w(n) = x(n) - \hat{g}_p y(n) - \hat{g}_c z(n). \quad (57)$$

Since the signals $x(n)$, $y(n)$, and $z(n)$ are available, the states of the weighting filter are updated by computing $e_w(n)$ as in Equation (54) for $n = 48,\ldots,63$. This saves two filterings.

5.11 High-band gain generation

In order to compute the high band gain for 23.85 kbit/s mode, 16 kHz input speech is filtered through a band-pass FIR filter $H_{HB}(z)$ which has the passband from 6.4 to 7 kHz. The high band gain g_{HB} is obtained by

$$g_{HB} = \frac{\sum_{i=0}^{63} (s_{HB}(i))^2}{\sum_{i=0}^{63} (s_{HB2}(i))^2}, \quad (58)$$

where $s_{HB}(i)$ is band-pass filtered input speech and $s_{HB}(i)$ is high-band excitation $u_{HB2}(i)$ filtered through high-band synthesis filter $A_{HB}(z)$.

6 Functional description of the decoder

The function of the decoder consists of decoding the transmitted parameters (LP parameters, adaptive codebook vector, adaptive codebook gain, fixed codebook vector, fixed codebook gain and high-band gain) and performing synthesis to obtain the reconstructed speech. The reconstructed speech is then postprocessed and upsampled (and upscaled). Finally high-band signal is generated to the frequency band from 6 to 7 kHz. The signal flow at the decoder is shown in Figure 3.

6.1 Decoding and speech synthesis

The decoding process is performed in the following order:

Decoding of LP filter parameters: The received indices of ISP quantization are used to reconstruct the quantized ISP vector. The interpolation described in Section 5.2.6 is performed to obtain 4 interpolated ISP vectors (corresponding to 4 subframes). For each subframe, the interpolated ISP vector is converted to LP filter coefficient domain a_k, which is used for synthesizing the reconstructed speech in the subframe.

The following steps are repeated for each subframe:

1. **Decoding of the adaptive codebook vector:** The received pitch index (adaptive codebook index) is used to find the integer and fractional parts of the pitch lag. The adaptive codebook vector $v(n)$ is found by interpolating the past excitation $u(n)$ (at the pitch delay) using the FIR filter described in Section 5.6. The received adaptive filter index is used to find out whether the filtered adaptive codebook is $v_1(n)= v(n)$ or $v_2(n) = 0.18v(n)+0.64v(n-1)+0.18v(n-2)$.

2. **Decoding of the innovative vector:** The received algebraic codebook index is used to extract the positions and amplitudes (signs) of the excitation pulses and to find the algebraic codevector $c(n)$. If the integer part of the pitch lag is less than the subframe size 64, the pitch sharpening procedure is applied which translates into modifying $c(n)$ by filtering it through the adaptive prefilter $F(z)$ which consists of two parts: a periodicity enhancement part $1/(1\text{-}0.85z^{-T})$ and a tilt part $(1 - \beta_1 z^{-1})$, where T is the integer part of the pitch lag and $\beta_1(n)$ is related to the voicing of the previous subframe and is bounded by [0.0,0.5].

3. **Decoding of the adaptive and innovative codebook gains:** The received index gives the fixed codebook gain correction factor $\hat{\gamma}$. The estimated fixed codebook gain g'_c is found as described in Section 5.8. First, the predicted energy for every subframe n is found by

$$\tilde{E}(n) = \sum_{i=1}^{4} b_i \hat{R}(n-i) \qquad (59)$$

and then the mean innovation energy is found by

$$E_i = 10 \log\left(\frac{1}{N}\sum_{i=0}^{N-1} c^2(i)\right) \qquad (60)$$

The predicted gain g'_c is found by

$$g'_c = 10^{0.05(\tilde{E}(n)+\bar{E}-E_i)} \qquad (61)$$

The quantized fixed codebook gain is given by

$$\hat{g}_c = \hat{\gamma} g'_c. \qquad (62)$$

4. **Computing the reconstructed speech:** The following steps are for n = 0, ..., 63. The total excitation is constructed by:

$$u(n) = \hat{g}_p v(n) + \hat{g}_c c(n), \qquad . \qquad (63)$$

Before the speech synthesis, a post-processing of excitation elements is performed

5. **Anti-sparseness processing (6.60, 8.85 kbit/s modes):** An adaptive anti-sparseness post-processing procedure is applied to the fixed codebook vector $c(n)$ in order to reduce perceptual artifacts arising from the sparseness of the algebraic fixed codebook vectors with only a few non-zero samples per subframe. The anti-sparseness processing consists of circular convolution of the fixed codebook vector with an impulse response. Three pre-stored impulse responses are used and a number *impNr*=0,1,2 is set to select one of them. A value of 2 corresponds to no modification, a value of 1 corresponds to medium modification, while a value of 0 corresponds to strong modification. The selection of the impulse response is performed adaptively from the adaptive and fixed codebook gains. The following procedure is employed:

   if $\hat{g}_p < 0.6$ then
   $impNr = 0;$
   else if $\hat{g}_p < 0.9$ then
   $impNr = 1;$
   else
   $impNr = 2;$

   Detect onset by comparing the fixed codebook gain to the previous fixed codebook gain. If the current value is more than three times the previous value an onset is detected.

   If not onset and *impNr*=0, the median filtered value of the current and the previous 4 adaptive codebook gains are computed. If this value is less than 0.6, *impNr*=0.

   If not onset, the *impNr*-value is restricted to increase by one step from the previous subframe.

   If an onset is declared, the *impNr* -value is increased by one if it is less than 2.

   In case of 8.85 kbit/s mode, the *impNr* -value is increased by one.

6. **Noise enhancer:** A nonlinear gain smoothing technique is applied to the fixed codebook gain $\hat{g}_c$ in order to enhance excitation on noise. Based on the stability and voicing of the speech segment, the gain of the fixed codebook is smoothed in order to reduce fluctuation in the energy of the excitation in case of stationary signals. This improves the performance in case of stationary background noise.

   The voicing factor is given by λ=0.5(1-r_v) with r_v=(E_v-E_c)/(E_v+E_c), where E_v and E_c are the energies of the scaled pitch codevector and scaled innovation codevector, respectively. Note that since the value of r_v is between –1 and 1, the value of λ is between 0 and 1. Note that the factor λ is related to the amount of unvoicing with a value of 0 for purely voiced segments and a value of 1 for purely unvoiced segments.

   A stability factor θ is computed based on a distance measure between the adjacent LP filters. Here, the factor θ is related to the ISP distance measure and it is bounded by $0 \le \theta \le 1$, with larger values of θ corresponding to more stable signals.

   Finally, the a gain smoothing factor S_m is given by

$$S_m = \lambda\theta. \qquad (64)$$

   The value of S_m approaches 1 for unvoiced and stable signals, which is the case of stationary background noise signals. For purely voiced signals or for unstable signals, the value of S_m approaches 0.

   An initial modified gain g_0 is computed by comparing the fixed codebook gain $\hat{g}_c$ to a threshold given by the initial modified gain from the previous subframe, g_{-1}. If $\hat{g}_c$ is larger or equal to g_{-1}, then g_0 is computed by decrementing $\hat{g}_c$ by 1.5 dB bounded by $g_0 \ge g_{-1}$. If $\hat{g}_c$ is smaller than g_{-1}, then g_0 is computed by incrementing $\hat{g}_c$ by 1.5 dB bounded by $g_0 \le g_{-1}$.

Finally, the gain is update with the value of the smoothed gain as follows

$$\hat{g}_c = \theta g_0 + (1-\theta)\hat{g}_c, \quad (65)$$

7. **Pitch enhancer:** A pitch enhancer procedure modifies the total excitation $u(n)$ by filtering the fixed codebook excitation through an innovation filter whose frequency response emphasizes the higher frequencies more than lower frequencies, and whose coefficients are related to the periodicity in the signal. A filter of the form

$$F_{inno}(z) = -c_{pe}z + 1 - c_{pe}z^{-1}, \quad (66)$$

where c_{pe}=0.125(1– r_v), with r_v=(E_v-E_c)/(E_v+E_c) as described above. The filtered fixed codevector is given by

$$c'(n) = c(n) - c_{pe}(c(n+1) + c(n-1)). \quad (67)$$

and the updated excitation is given by

$$u(n) = \hat{g}_p v(n) + \hat{g}_c c'(n). \quad (68)$$

The above procedure can be done in one step by updating the excitation as follows

$$u(n) = u(n) - \hat{g}_c c_{pe}(c(n+1) + c(n-1)). \quad (69)$$

8. **Post-processing of excitation elements (6.60 kbit/s mode):** A post-processing of excitation elements procedure is applied to the total excitation $u(n)$ by emphasizing the contribution of the adaptive codebook vector:

$$\hat{u}(n) = \begin{cases} u(n) + 0.25\beta\hat{g}_p v(n), & \hat{g}_p > 0.5 \\ u(n), & \hat{g}_p \le 0.5 \end{cases} \quad (70)$$

Adaptive gain control (AGC) is used to compensate for the gain difference between the non-emphasized excitation $u(n)$ and emphasized excitation $\hat{u}(n)$ The gain scaling factor η for the emphasized excitation is computed by:

$$\eta = \begin{cases} \sqrt{\dfrac{\sum_{n=0}^{63} u^2(n)}{\sum_{n=0}^{63} \hat{u}^2(n)}}, & \hat{g}_p > 0.5, \\ 1.0, & \hat{g}_p \le 0.5. \end{cases} \quad (71)$$

The gain-scaled emphasized excitation signal $\hat{u}'(n)$ is given by:

$$\hat{u}'(n) = \hat{u}(n)\eta. \quad (72)$$

The reconstructed speech for the subframe of size 64 is given by

$$\hat{s}(n) = \hat{u}(n) - \sum_{i=1}^{16} \hat{a}_i \hat{s}(n-i), \qquad n = 0,\ldots,63. \quad (73)$$

where $\hat{a}_i$ are the interpolated LP filter coefficients.

The synthesis speech $\hat{s}(n)$ is then passed through an adaptive postprocessing which is described in the following section.

6.2 High-pass filtering, up-scaling and interpolation

The high-pass filter serves as a precaution against undesired low frequency components. The signal is filtered through the high-pass filter $H_{h1}(z)$ and de-emphasis filter $H_{de\_emph}(z)$.

Finally, the signal is upsampled to 16 kHz to obtain the lower band synthesis $\hat{s}_{16k}$. $\hat{s}_{16k}$ is produced by first upsampling the lower band synthesis $\hat{s}_{12.8k}$ at 12.8 kHz by 5, then filtering the output through $H_{decim}(z)$, and finally downsampling it by 4.

(Up-scaling consists of multiplying the output from the high-pass filtering by a factor of 2 in order to compensate the down-scaling at the pre-processing stage.)

6.3 High frequency band

For higher frequency band (6.4 – 7.0 kHz), excitation is generated to model the highest frequencies. The signal is generated to the frequency band between 6.4 and 7.0 kHz. The high frequency content is generated by filling the upper part of the spectrum with a white noise properly scaled in the excitation domain, then converted to the speech domain by shaping it with a filter derived from the same LP synthesis filter used for synthesizing the down-sampled signal.

6.3.1 Generation of high-band excitation

This high-band excitation is obtained by first generating white noise $u_{HB1}(n)$. The power of the high-band excitation is set equal to the power of excitation $u_2(n)$ which means that

$$u_{HB2}(n) = u_{HB1}(n) \sqrt{\sum_{k=0}^{63} {u_2}^2(k) \Big/ \sum_{k=0}^{63} {u_{HB1}}^2(k)} \; . \qquad (74)$$

Finally the high-band excitation is found by

$$u_{HB}(n) = \hat{g}_{HB} u_{HB2}(n) \Big/ \sqrt{\sum_{k=0}^{63} {u_{HB2}}^2(k)} \; , \qquad (75)$$

where is $\hat{g}_{HB}$ is a gain factor.

In the 23.85 kbit/s mode, $\hat{g}_{HB}$ is decoded from the received gain index.

In 6.60, 8.85, 12.65, 14.25, 15.85, 18.25, 19.85 or 23.05 kbit/s modes, g_{HB} is estimated using voicing information bounded by [0.1,1.0]. First, tilt of synthesis e_{tilt} is found

$$e_{tilt} = \sum_{n=0}^{63} \hat{s}(n)\hat{s}(n-1) \Big/ \sum_{n=1}^{63} \hat{s}^2(n) \qquad (76)$$

conditioned by $e_{tilt} \geq 0$ and $e_{tilt} \geq r_v$, with r_v being the voicing factor defined above. The $\hat{g}_{HB}$ is then found by

$$g_{HB} = w_{SP} g_{SP} + (1 - w_{SP}) g_{BG} \; , \qquad (77)$$

where g_{SP} = 1 - e_{tilt} is gain for speech signal, g_{BG} = 1.25g_{SP} gain for background noise signal, and w_{SP} is a weighting function which is 1, when VAD is ON, and 0 when VAD is OFF.

The tilt factor is introduced into the gain computation to take into account the high frequency contents of the synthesized signal. In case of voiced segments where less energy is present at high frequencies, e_{tilt} approaches 1 resulting in a lower gain g_{HB}. This reduces the energy of the generated noise in case of voiced segments.

6.3.2 LP filter for the high frequency band

6.3.2.1 6.60 kbit/s mode

The high-band LP synthesis filter $A_{HB}(z)$ is found by extrapolating the quantized ISF vector **f** into 20th order ISF vector $\mathbf{f}_e$. First, maximum of the autocorrelation $C_{max}(i)$ of ISF vector difference vector $f_\Delta(i) = f(i+1) - f(i), i = 1,...,14$ is obtained. Then new 16kHz ISF vector $f_e'(i)$ is computed by

$$f_e'(i) = \begin{cases} f(i-1), & i = 1,..,15 \\ f_e'(i-1) + f_e'(i - C_{\max}(i) - 1) - f_e'(i - C_{\max}(i) - 2), & i = 16,..,19 \end{cases}. \quad (78)$$

An approximation of the last element of new ISF vector f_{e19} is updated based on lower frequency coefficients. New extrapolated ISF vector difference vector $f_{e\Delta}'(i)$ is

$$f_{e\Delta}'(i) = c_{scale}\left(f_e'(i) - f_e'(i-1)\right), i = 16,...,19, \quad (79)$$

where c_{scale} scales $f_{e\Delta}'(i)$ so that $f_e(19)$ will be equal to f_{e19}. In order to insure stability, $f_{e\Delta}'(i)$ is bounded by

$$f_{e\Delta}'(i) + f_{e\Delta}'(i-1) > 500, i = 17,...,19. \quad (80)$$

Finally the extrapolated ISF vector $\mathbf{f}_e$ is obtained by

$$f_e(i) = \begin{cases} f(i), & i = 1,...,15 \\ f_{e\Delta}'(i) + f_e(i-1), & i = 16,...,19 \\ f(16), & i = 20 \end{cases}. \quad (81)$$

$\mathbf{f}_e$ is converted to cosine domain to obtain $\mathbf{q}_e$ with 16000 Hz sampling rate. The high-band LP synthesis filter $A_{HB}(z)$ is obtained by converting $\mathbf{q}_e$ to LP filter as described in 5.2.4 with m=20.

6.3.2.2 8.85, 12.65, 14.25, 15.85, 18.25, 19.85, 23.05 or 23.85 kbit/s modes

The high-band LP synthesis filter $A_{HB}(z)$ is weighted low-band LP synthesis filter

$$A_{HB}(z) = \hat{A}(z/0.8), \quad (82)$$

where $\hat{A}(z)$ is the interpolated LP synthesis filter. $\hat{A}(z)$ has been computed analysing signal with the sampling rate of 12.8 kHz but it is now used for a 16 kHz signal. Effectively, this means that the frequency response $FR_{16}(f)$ of $A_{HB}(z)$ is obtained by

$$FR_{16}(f) = FR_{12.8}(\frac{12.8}{16} f), \quad (83)$$

where $FR_{12.8}(f)$ is the frequency response of $A(z)$. This mean that the band 5.1 – 5.6 kHz in 12.8 kHz domain will be mapped to 6.4 – 7.0 kHz in 16 kHz domain.

6.3.3 High band synthesis

$u_{HB}(n)$ is filtered through $A_{HB}(z)$. The output of this high-band synthesis $s_{HB}(n)$ is filtered through a band-pass FIR filter $H_{HB}(z)$ which has the passband from 6 to 7 kHz. Finally, s_{HB} is added to synthesized speech $\hat{s}_{16k}$ to produce the synthesized output speech signal $\hat{s}_{output}$.

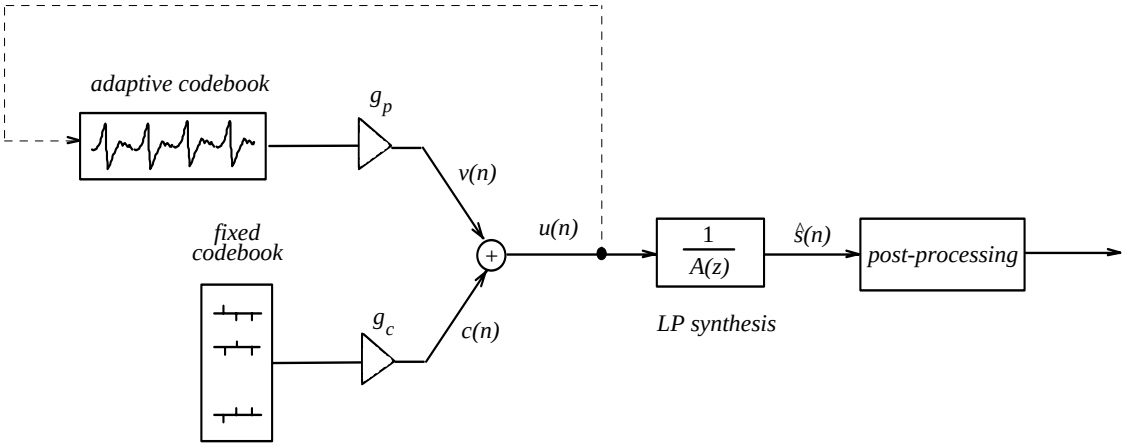


Figure 1 Simplified block diagram of the CELP synthesis model

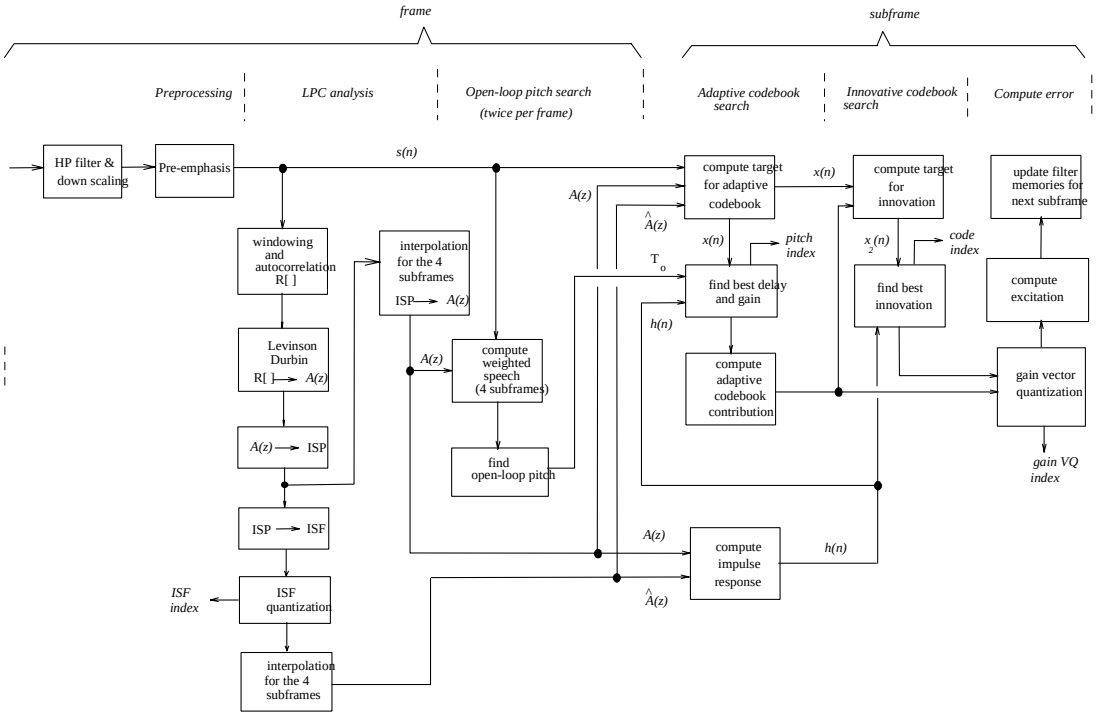


Figure 2 Detailed block diagram of the ACELP encoder

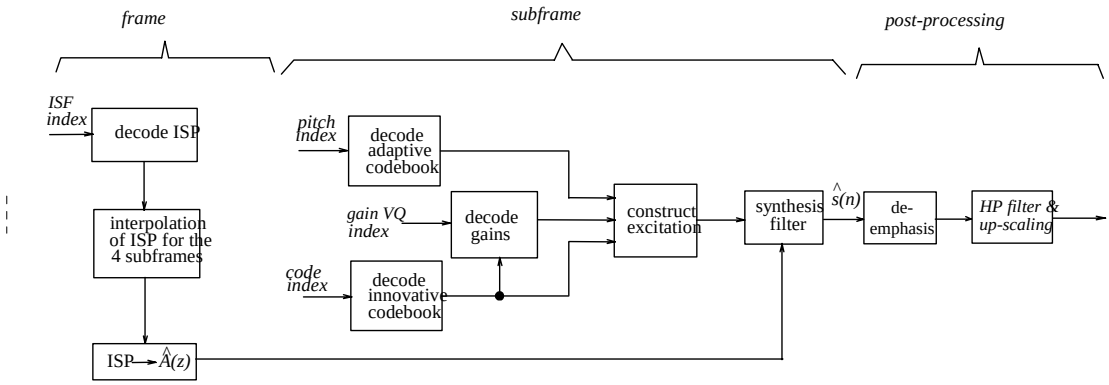


Figure 3 Detailed block diagram of the ACELP decoder

7 Detailed bit allocation of the adaptive multi-rate wideband codec

The detailed allocation of the bits in the adaptive multi-rate wideband speech encoder is shown for each mode in table 12a-12i. These tables show the order of the bits produced by the speech encoder. Note that the most significant bit (MSB) of each codec parameter is always sent first.

Table 12a: Source encoder output parameters in order of occurrence and bit allocation within the speech frame of 477 bits/20 ms, 23.85 kbit/s mode.

Bits (MSB-LSB)	Description
s1	VAD-flag
s2 – s9	index of 1st ISP subvector
s10 – s17	index of 2nd ISP subvector
s18 - s23	index of 3rd ISP subvector
s24 – s30	index of 4th ISP subvector
s31 – s37	index of 5th ISP subvector
s38 – s42	index of 6th ISP subvector
s43 – s47	index of 7th ISP subvector
subframe 1	
s48 – s56	adaptive codebook index
s57	LTP-filtering-flag
s58 – s68	Codebook Index1 for track 1
s69 – s79	Codebook Index1 for track 2
ss80 –s90	Codebook Index1 for track 3
s91-s101	Codebook Index1 for track 4
s102-s112	Codebook Index2 for track 1
s113-s123	Codebook Index2 for track 2
s124 – s134	Codebook Index2 for track 3
s135 – s145	Codebook Index2 for track 4
s146 – s152	codebook gains
s153 – s156	High-band energy
subframe 2	
s157 – s162	adaptive codebook index (relative)
s163 – s262	same description as s57 – s156
subframe 3	
s263 – s371	same description as s48 – s156
subframe 4	
s372 – s476	same description as s157 – s262

Table 12b: Source encoder output parameters in order of occurrence and bit allocation within the speech frame of 461 bits/20 ms, 23.05 kbit/s mode.

Bits (MSB-LSB)	Description
s1	VAD-flag
s2 – s9	index of 1st ISP subvector
s10 – s17	index of 2nd ISP subvector
s18 - s23	index of 3rd ISP subvector
s24 – s30	index of 4th ISP subvector
s31 – s37	index of 5th ISP subvector
s38 – s42	index of 6th ISP subvector
s43 – s47	index of 7th ISP subvector
subframe 1	
s48 – s56	adaptive codebook index
s57	LTP-filtering-flag
s58 – s68	Codebook Index1 for track 1
s69 – s79	Codebook Index1 for track 2
ss80 –s90	Codebook Index1 for track 3
s91-s101	Codebook Index1 for track 4
s102-s112	Codebook Index2 for track 1
s113-s123	Codebook Index2 for track 2
s124 – s134	Codebook Index2 for track 3
s135 – s145	Codebook Index2 for track 4
s146 – s152	codebook gains
subframe 2	
s153 – s158	adaptive codebook index (relative)
s159 – s254	same description as s57 – s152
subframe 3	
s255 – s359	same description as s48 – s152
subframe 4	
s360 – s461	same description as s153 – s254

Table 12c: Source encoder output parameters in order of occurrence and bit allocation within the speech frame of 397 bits/20 ms, 19.85 kbit/s mode.

Bits (MSB-LSB)	Description
s1	VAD-flag
s2 – s9	index of 1st ISP subvector
s10 – s17	index of 2nd ISP subvector
s18 - s23	index of 3rd ISP subvector
s24 – s30	index of 4th ISP subvector
s31 – s37	index of 5th ISP subvector
s38 – s42	index of 6th ISP subvector
s43 – s47	index of 7th ISP subvector
subframe 1	
s48 – s56	adaptive codebook index
s57	LTP-filtering-flag
s58 – s67	Codebook Index1 for track 1
s68 – s77	Codebook Index1 for track 2
s78 – s79	Pulse Selector for track 3
s80 – s81	Pulse Selector for track 4
s82 – s91	Codebook index2 for track 1
s92 – s101	Codebook index2 for track 2
s102 – s115	Codebook index for track 3
s116 – s129	Codebook index for track 4
s130 – s136	VQ gain
subframe 2	
s137 – s142	adaptive codebook index (relative)
s143 – s222	same description as s57 – s136
subframe 3	
s223 – s311	same description as s48 – s136
subframe 4	
s312 – s397	same description as s137 – s222

Table 12d: Source encoder output parameters in order of occurrence and bit allocation within the speech frame of 365 bits/20 ms, 18.25 kbit/s mode.

Bits (MSB-LSB)	Description
s1	VAD-flag
s2 – s9	index of 1st ISP subvector
s10 – s17	index of 2nd ISP subvector
s18 - s23	index of 3rd ISP subvector
s24 – s30	index of 4th ISP subvector
s31 – s37	index of 5th ISP subvector
s38 – s42	index of 6th ISP subvector
s43 – s47	index of 7th ISP subvector
subframe 1	
s48 – s56	adaptive codebook index
s57	LTP-filtering-flag
s58 – s59	Pulse Selector for track 1
s60 – s61	Pulse Selector for track 2
s62 – s63	Pulse Selector for track 3
s64 – s65	Pulse Selector for track 4
s66 – s79	Codebook index for track 1
s80 – s93	Codebook index for track 2
s94 – s107	Codebook index for track 3
s108 – s121	Codebook index for track 4
s122 – s128	VQ gain
subframe 2	
s129 – s134	adaptive codebook index (relative)
s135 – s206	same description as s57 – s128
subframe 3	
s207 – s287	same description as s48 – s128
subframe 4	
s288 – s365	same description as s129 – s206

Table 12e: Source encoder output parameters in order of occurrence and bit allocation within the speech frame of 317 bits/20 ms, 15.85 kbit/s mode.

Bits (MSB-LSB)	Description
s1	VAD-flag
s2 – s9	index of 1st ISP subvector
s10 – s17	index of 2nd ISP subvector
s18 - s23	index of 3rd ISP subvector
s24 – s30	index of 4th ISP subvector
s31 – s37	index of 5th ISP subvector
s38 – s42	index of 6th ISP subvector
s43 – s47	index of 7th ISP subvector
subframe 1	
s48 – s56	adaptive codebook index
s57	LTP-filtering-flag
s58 – s70	Codebook index for track 1
s71 – s83	Codebook index for track 2
s84 – s96	Codebook index for track 3
s97 – s109	Codebook index for track 4
s110 – s116	VQ gain
subframe 2	
s117 – s122	adaptive codebook index (relative)
s123 – s182	same description as s57 – s116
subframe 3	
s183 – s251	same description as s48 – s116
subframe 4	
s252 – s317	same description as s117 – s182

Table 12f: Source encoder output parameters in order of occurrence and bit allocation within the speech frame of 285 bits/20 ms, 14.25 kbit/s mode.

Bits (MSB-LSB)	Description
s1	VAD-flag
s2 – s9	index of 1st ISP subvector
s10 – s17	index of 2nd ISP subvector
s18 - s23	index of 3rd ISP subvector
s24 – s30	index of 4th ISP subvector
s31 – s37	index of 5th ISP subvector
s38 – s42	index of 6th ISP subvector
s43 – s47	index of 7th ISP subvector
subframe 1	
s48 – s56	adaptive codebook index
s57	LTP-filtering-flag
s58 – s70	Codebook index for track 1
s71 – s83	Codebook index for track 2
s84 – s92	Codebook index for track 3
s93 – s101	Codebook index for track 4
s102 – s108	VQ gain
subframe 2	
s109 – s114	adaptive codebook index (relative)
s115 – s166	same description as s57 – s108
subframe 3	
s167 – s227	same description as s48 – s108
subframe 4	
s228 – s285	same description as s109 – s166

Table 12g: Source encoder output parameters in order of occurrence and bit allocation within the speech frame of 253 bits/20 ms, 12.65 kbit/s mode.

Bits (MSB-LSB)	Description
s1	VAD-flag
s2 – s9	index of 1st ISP subvector
s10 – s17	index of 2nd ISP subvector
s18 - s23	index of 3rd ISP subvector
s24 – s30	index of 4th ISP subvector
s31 – s37	index of 5th ISP subvector
s38 – s42	index of 6th ISP subvector
s43 – s47	index of 7th ISP subvector
subframe 1	
s48 – s56	adaptive codebook index
s57	LTP-filtering-flag
s58 – s66	Codebook index for track 1
s67 – s75	Codebook index for track 2
s76 – s84	Codebook index for track 3
s85 – s93	Codebook index for track 4
s94 – s100	VQ gain
subframe 2	
s101 – s106	adaptive codebook index (relative)
s107 – s150	same description as s57 – s100
subframe 3	
s151 – s203	same description as s48 – s100
subframe 4	
s204 – s253	same description as s101 – s150

Table 12h: Source encoder output parameters in order of occurrence and bit allocation within the speech frame of 177 bits/20 ms, 8.85 kbit/s mode.

Bits (MSB-LSB)	Description
s1	VAD-flag
s2 – s9	index of 1st ISP subvector
s10 – s17	index of 2nd ISP subvector
s18 - s23	index of 3rd ISP subvector
s24 – s30	index of 4th ISP subvector
s31 – s37	index of 5th ISP subvector
s38 – s42	index of 6th ISP subvector
s43 – s47	index of 7th ISP subvector
subframe 1	
s48 – s55	adaptive codebook index
s56 – s60	Codebook index for track 1
s61 – s65	Codebook index for track 2
s66 – s70	Codebook index for track 3
s71 - s75	Codebook index for track 4
s76 – s81	VQ gain
subframe 2	
s82 – s86	adaptive codebook index (relative)
s87 – s112	same description as s56 – s81
subframe 3	
s113 – s146	same description as s48 – s81
subframe 4	
s147 – s177	same description as s82 – s112

Table 12i: Source encoder output parameters in order of occurrence and bit allocation within the speech frame of 132 bits/20 ms, 6.60 kbit/s mode.

Bits (MSB-LSB)	Description
s1	VAD-flag
s2 – s9	index of 1st ISP subvector
s10 – s17	index of 2nd ISP subvector
s18 – s24	index of 3rd ISP subvector
s25 – s31	index of 4th ISP subvector
s32 – s37	index of 5th ISP subvector
subframe 1	
s38 – s45	adaptive codebook index
s46 - 57	Codebook Index
s58 – s63	VQ gain
subframe 2	
s64 – s68	adaptive codebook index (relative)
s69 – s86	same description as s46 – s63
subframe 3	
s87 – s109	same description as s64 – s86
subframe 4	
s110 – s132	same description as s64 – s86

8 Bibliography

1) M.R. Schroeder and B.S. Atal, "Code-Excited Linear Prediction (CELP): High quality speech at very low bit rates," in Proc. *ICASSP'85*, pp. 937-940, 1985.

2) L.R. Rabiner and R.W. Schaefer. *Digital processing of speech signals*. Prentice-Hall Int., 1978.

3) F. Itakura, "Line spectral representation of linear predictive coefficients of speech signals," *J. Acoust. Soc. Amer.*, vol. 57, Supplement no. 1, S35, 1975.

4) F.K. Soong and B.H. Juang, "Line spectrum pair (LSP) and speech data compression", in *Proc. ICASSP'84*, pp. 1.10.1-1.10.4.

5) K.K Paliwal and B.S. Atal, "Efficient vector quantization of LPC parameters at 24 bits/frame", *IEEE Trans. Speech and Audio Processing*, vol. 1, no 1, pp. 3-14, 1993.

6) P. Kabal and R.P. Ramachandran, "The computation of line spectral frequencies using Chebyshev polynomials", *IEEE Trans. on ASSP*, vol. 34, no. 6, pp. 1419-1426, Dec. 1986.

7) K. Järvinen, J. Vainio, P. Kapanen, T. Honkanen, P. Haavisto, R. Salami, C. Laflamme, and J.-P. Adoul, "GSM enhanced full rate speech codec", in *Proc. ICASSP'97*, pp. 771-774.

8) T. Honkanen, J. Vainio, K. Järvinen, P. Haavisto, R. Salami, C. Laflamme, and J.-P. Adoul, "Enhanced full rate speech codec for IS-136 digital cellular system", in *Proc. ICASSP'97*, pp. 731-734.

9) R. Hagen, E. Ekudden, B. Johansson, and W.B. Kleijn, "Removal of sparse-excitation artifacts in CELP", in *Proc. ICASSP'98*, pp. I-145-I-148.

Annex A (informative): Change history

Change history							
Date	**TSG #**	**TSG Doc.**	**CR**	**Rev**	**Subject/Comment**	**Old**	**New**
12-2000	10	SP-000558			Version 1.0.0 provided for information		