3GPP TS 26.090 V18.0.0 (2024-03)

Technical Specification

3rd Generation Partnership Project;

Technical Specification Group Services and System Aspects;

Mandatory Speech Codec speech processing functions;

Adaptive Multi-Rate (AMR) speech codec;

Transcoding functions

(Release 18)

** 

The present document has been developed within the 3rd Generation Partnership Project (3GPP TM) and may be further elaborated for the purposes of 3GPP.  
The present document has not been subject to any approval process by the 3GPPOrganizational Partners and shall not be implemented.  
This Specification is provided for future development work within 3GPPonly. The Organizational Partners accept no liability for any use of this Specification.  
Specifications and reports for implementation of the 3GPP TM system should be obtained via the 3GPP Organizational Partners' Publications Offices.

Keywords

GSM, UMTS, codec, LTE

***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

***Copyright Notification***

No part may be reproduced except as authorized by written permission.  
The copyright and the foregoing restriction extend to reproduction in all media.

© 2024, 3GPP Organizational Partners (ARIB, ATIS, CCSA, ETSI, TSDSI, TTA, TTC).

All rights reserved.

UMTS™ is a Trade Mark of ETSI registered for the benefit of its members

3GPP™ is a Trade Mark of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners  
LTE™ is a Trade Mark of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners

GSM® and the GSM logo are registered and owned by the GSM Association

Contents

Foreword 6

1 Scope 7

2 References 7

3 Definitions, symbols and abbreviations 7

3.1 Definitions 7

3.2 Symbols 9

3.3 Abbreviations 12

4 Outline description 13

4.1 Functional description of audio parts 13

4.2 Preparation of speech samples 14

4.2.1 PCM format conversion 14

4.3 Principles of the adaptive multi-rate 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 (all modes) 17

5.2 Linear prediction analysis and quantization 17

5.2.1 Windowing and auto**‑**correlation computation 18

5.2.2 Levinson**‑**Durbin algorithm (all modes) 19

5.2.3 LP to LSP conversion (all modes) 19

5.2.4 LSP to LP conversion (all modes) 21

5.2.5 Quantization of the LSP coefficients 21

5.2.6 Interpolation of the LSPs 23

5.2.7 Monitoring resonance in the LPC spectrum (all modes) 24

5.3 Open‑loop pitch analysis 24

5.4 Impulse response computation (all modes) 28

5.5 Target signal computation (all modes) 28

5.6 Adaptive codebook 28

5.6.1 Adaptive codebook search 28

5.6.2 Adaptive codebook gain control (all modes) 32

5.7 Algebraic codebook 32

5.7.1 Algebraic codebook structure 32

5.7.2 Algebraic codebook search 35

5.8 Quantization of the adaptive and fixed codebook gains 38

5.8.1 Adaptive codebook gain limitation in quantization 38

5.8.2 Quantization of codebook gains 38

5.8.3 Update past quantized adaptive codebook gain buffer (all modes) 41

5.9 Memory update (all modes) 41

6 Functional description of the decoder 41

6.1 Decoding and speech synthesis 41

6.2 Post‑processing 44

6.2.1 Adaptive post‑filtering (all modes) 44

6.2.2 High-pass filtering and up-scaling (all modes) 45

7 Detailed bit allocation of the adaptive multi-rate codec 46

8 Homing sequences 50

8.1 Functional description 50

8.2 Definitions 51

8.3 Encoder homing 51

8.4 Decoder homing 51

9 Bibliography 55

Annex A (informative): Change history 56

# Foreword

This Technical Specification has been produced by the 3rd Generation Partnership Project (3GPP).

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 the present document, 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 or greater 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 document.

# 1 Scope

The present document describes the detailed mapping from input blocks of 160 speech samples in 13‑bit uniform PCM format to encoded blocks of 95, 103, 118, 134, 148, 159, 204, and 244 bits and from encoded blocks of 95, 103, 118, 134, 148, 159, 204, and 244 bits to output blocks of 160 reconstructed speech samples. The sampling rate is 8 000 samples/s leading to a bit rate for the encoded bit stream of 4.75, 5.15, 5.90, 6.70, 7.40, 7.95, 10.2 or 12.2 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 multi-rate ACELP coder is referred to as MR-ACELP.

In the case of discrepancy between the requirements described in the present document and the fixed point computational description (ANSI‑C code) of these requirements contained in [4], the description in [4] will prevail. The ANSI‑C code is not described in the present document, see [4] for a description of the ANSI‑C code.

The transcoding procedure specified in the present document is mandatory for systems using the AMR speech codec.

# 2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication, edition number, version number, etc.) or non‑specific.

- For a specific reference, subsequent revisions do not apply.

- For a non-specific reference, the latest version applies. In the case of a reference to a 3GPP document (including a GSM document), a non-specific reference implicitly refers to the latest version of that document *in the same Release as the present document*.

[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] 3GPP TS 26.101 : "Frame Structure".

[3] 3GPP TS 26.094: "AMR Speech Codec; Voice Activity Detector".

[4] 3GPP TS 26.073: "Adaptive Multi-Rate (AMR); ANSI C source code".

[5] 3GPP TS 26.074: "Adaptive Multi-Rate (AMR); Test sequences".

[6] ITU‑T Recommendation G.711 (1988): "Pulse code modulation (PCM) of voice frequencies".

[7] ITU‑T Recommendation G.726: "40, 32, 24, 16 kbit/s Adaptive Differential Pulse Code Modulation (ADPCM)".

[8] ITU-T Recommendation G.712

# 3 Definitions, symbols and abbreviations

## 3.1 Definitions

For the purposes of the present document, the following terms and definitions apply:

**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

**adaptive postfilter:** this filter is applied to the output of the short-term synthesis filter to enhance the perceptual quality of the reconstructed speech. In the adaptive multi-rate codec, the adaptive postfilter is a cascade of two filters: a formant postfilter and a tilt compensation filter

**algebraic codebook:** fixed codebookwhere 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 positions

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

**closed‑loop pitch analysis:** 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 1/6th or 1/3rd of a sample is used.

**frame:** time interval equal to 20 ms (160 samples at an 8 kHz sampling rate)

**integer lags:** set of lag values having whole sample resolution

**interpolating filter:** FIR filter used to produce an estimate of subsample resolution samples, given an input sampled with integer sample resolution

**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:** long term filter delay. This is typically the true pitch period, or its multiple or sub‑multiple

**Line Spectral Frequencies:** (see Line Spectral Pair)

**Line Spectral Pair:** transformation of LPC parameters. Line 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 Line Spectral Pairs (also called as Line Spectral Frequencies) are the roots of these polynomials on the z-unit circle

**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 240 samples. For each frame, two asymmetric windows are used to generate two sets of LP coefficient in the 12.2 kbit/s mode. For the other modes, only a single asymmetric window is used to generate a single set of LP coefficients. In the 12.2 kbit/s mode, no samples of the future frames are used (no lookahead). The other modes use a 5 ms lookahead

**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 codec

**open‑loop pitch search:** 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:** thisfilter 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:** time interval equal to 5 ms (40 samples at 8 kHz sampling rate)

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

**zero input response:** 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:** 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 the present document, the following symbols apply:

 The inverse filter with unquantized coefficients

 The inverse filter with quantized coefficients

 The speech synthesis filter with quantized coefficients

 The unquantized linear prediction parameters (direct form coefficients)

 The quantified linear prediction parameters

 The order of the LP model

 The long‑term synthesis filter

 The perceptual weighting filter (unquantized coefficients)

 The perceptual weighting factors

 Adaptive pre‑filter

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

 The adaptive pre‑filter coefficient (the quantified pitch gain)

 The formant postfilter

 Control coefficient for the amount of the formant post‑filtering

 Control coefficient for the amount of the formant post‑filtering

 Tilt compensation filter

 Control coefficient for the amount of the tilt compensation filtering

 A tilt factor, with being the first reflection coefficient

 The truncated impulse response of the formant postfilter

 The length of 

 The auto‑correlations of 

 The inverse filter (numerator) part of the formant postfilter

 The synthesis filter (denominator) part of the formant postfilter

 The residual signal of the inverse filter 

 Impulse response of the tilt compensation filter

 The AGC‑controlled gain scaling factor of the adaptive postfilter

 The AGC factor of the adaptive postfilter

 Pre‑processing high‑pass filter

,  LP analysis windows

 Length of the first part of the LP analysis window 

 Length of the second part of the LP analysis window 

 Length of the first part of the LP analysis window 

 Length of the second part of the LP analysis window 

 The auto‑correlations of the windowed speech 

 Lag window for the auto‑correlations (60 Hz bandwidth expansion)

 The bandwidth expansion in Hz

 The sampling frequency in Hz

 The modified (bandwidth expanded) auto‑correlations

 The prediction error in the *i*th iteration of the Levinson algorithm

 The *i*th reflection coefficient

 The *j*th direct form coefficient in the *i*th iteration of the Levinson algorithm

 Symmetric LSF polynomial

 Antisymmetric LSF polynomial

 Polynomial  with root  eliminated

 Polynomial  with root  eliminated

 The line spectral pairs (LSPs) in the cosine domain

 An LSP vector in the cosine domain

 The quantified LSP vector at the *i*th subframe of the frame *n*

 The line spectral frequencies (LSFs)

 A th order Chebyshev polynomial

 The coefficients of the polynomials and 

 The coefficients of the polynomials  and 

 The coefficients of either  or 

 Sum polynomial of the Chebyshev polynomials

 Cosine of angular frequency 

 Recursion coefficients for the Chebyshev polynomial evaluation

 The line spectral frequencies (LSFs) in Hz

 The vector representation of the LSFs in Hz

,  The mean‑removed LSF vectors at frame *n*

,  The LSF prediction residual vectors at frame *n*

 The predicted LSF vector at frame *n*

 The quantified second residual vector at the past frame

 The quantified LSF vector at quantization index *k*

 The LSP quantization error

 LSP‑quantization weighting factors

 The distance between the line spectral frequencies  and 

 The impulse response of the weighted synthesis filter

 The correlation maximum of open‑loop pitch analysis at delay *k*

 The correlation maxima at delays 

 The normalized correlation maxima  and the corresponding delays 

 The weighted synthesis filter

 The numerator of the perceptual weighting filter

 The denominator of the perceptual weighting filter

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

 The windowed speech signal

 The weighted speech signal

 Reconstructed speech signal

 The gain‑scaled post‑filtered signal

 Post‑filtered speech signal (before scaling)

 The target signal for adaptive codebook search

,  The target signal for algebraic codebook search

 The LP residual signal

 The fixed codebook vector

 The adaptive codebook vector

 The filtered adaptive codebook vector

 The past filtered excitation

 The excitation signal

 The emphasized adaptive codebook vector

 The gain‑scaled emphasized excitation signal

 The best open‑loop lag

 Minimum lag search value

 Maximum lag search value

 Correlation term to be maximized in the adaptive codebook search

 The FIR filter for interpolating the normalized correlation term 

 The interpolated value of  for the integer delay *k* and fraction *t*

 The FIR filter for interpolating the past excitation signal  to yield the adaptive codebook vector 

 Correlation term to be maximized in the algebraic codebook search at index *k*

 The correlation in the numerator of  at index *k*

 The energy in the denominator of  at index *k*

 The correlation between the target signal  and the impulse response , i.e., backward filtered target

 The lower triangular Toepliz convolution matrix with diagonal  and lower diagonals 

 The matrix of correlations of 

 The elements of the vector **d**

 The elements of the symmetric matrix 

 The innovation vector

 The correlation in the numerator of 

 The position of the *i*th pulse

 The amplitude of the *i*th pulse

 The number of pulses in the fixed codebook excitation

 The energy in the denominator of 

 The normalized long‑term prediction residual

 The signal used for presetting the signs in algebraic codebook search

 The sign signal for the algebraic codebook search

 Sign extended backward filtered target

 The modified elements of the matrix , including sign information

,  The fixed codebook vector convolved with 

 The mean‑removed innovation energy (in dB)

 The mean of the innovation energy

 The predicted energy

 The MA prediction coefficients

 The quantified prediction error at subframe *k*

 The mean innovation energy

 The prediction error of the fixed‑codebook gain quantization

 The quantization error of the fixed‑codebook gain quantization

 The states of the synthesis filter 

 The perceptually weighted error of the analysis‑by‑synthesis search

 The gain scaling factor for the emphasized excitation

 The fixed‑codebook gain

 The predicted fixed‑codebook gain

 The quantified fixed codebook gain

 The adaptive codebook gain

 The quantified adaptive codebook gain

 A correction factor between the gain  and the estimated one 

 The optimum value for 

 Gain scaling factor

## 3.3 Abbreviations

For the purposes of the present document, the following abbreviations apply.

ACELP Algebraic Code Excited Linear Prediction

AGC Adaptive Gain Control

AMR Adaptive Multi-Rate

CELP Code Excited Linear Prediction

EFR Enhanced Full Rate

FIR Finite Impulse Response

ISPP Interleaved Single‑Pulse Permutation

LP Linear Prediction

LPC Linear Predictive Coding

LSF Line Spectral Frequency

LSP Line Spectral Pair

LTP Long Term Predictor (or Long Term Prediction)

MA Moving Average

# 4 Outline description

The present document is structured as follows:

Clause 4.1 contains a functional description of the audio parts including the A/D and D/A functions. Clause 4.2 describes the conversion between 13‑bit uniform and 8‑bit A‑law or -law samples. Clauses 4.3 and 4.4 present a simplified description of the principles of the AMR codec encoding and decoding process respectively. In clause 4.5, the sequence and subjective importance of encoded parameters are given.

Clause 5 presents the functional description of the AMR codec encoding, whereas clause 6 describes the decoding procedures. In clause 7, the detailed bit allocation of the AMR 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 8 kHz;

- analogue‑to‑uniform digital conversion to 13‑bit representation.

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

2) Uniform digital PCM to analogue

- conversion from 13‑bit/8 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 either:

- by direct conversion to 13‑bit uniform PCM format;

- or by conversion to 8‑bit A‑law or -law compounded format, based on a standard A‑law or -law codec/filter according to ITU‑T Recommendations G.711 [6] and G.714, followed by the 8‑bit to 13‑bit conversion as specified in clause 4.2.1.

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

In the latter case it should be noted that the specifications in ITU‑T G.714 (superseded by G.712) are concerned with PCM equipment located in the central parts of the network. When used in the terminal equipment, the present document does not on its own ensure sufficient out‑of‑band attenuation. The specification of out‑of‑band signals is defined in [1] in clause 2.

## 4.2 Preparation of speech samples

The encoder is fed with data comprising of samples with a resolution of 13 bits left justified in a 16‑bit word. The three least significant bits are set to '0'. 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.

### 4.2.1 PCM format conversion

The conversion between 8‑bit A‑Law or -law compressed data and linear data with 13‑bit resolution at the speech encoder input shall be as defined in ITU‑T Rec. G.711 [6].

ITU‑T Rec. G.711 [6] specifies the A‑Law or -law to linear conversion and vice versa by providing table entries. Examples on how to perform the conversion by fixed‑point arithmetic can be found in ITU‑T Rec. G.726 [7]. Clause 4.2.1 of G.726 [7] describes A‑Law or -law to linear expansion and clause 4.2.8 of G.726 [7] provides a solution for linear to A‑Law or -law compression.

## 4.3 Principles of the adaptive multi-rate speech encoder

The AMR codec consists of eight source codecs with bit-rates of 12.2, 10.2, 7.95, 7.40, 6.70, 5.90, 5.15 and 4.75 kbit/s.

The codec is based on the code‑excited linear predictive (CELP) coding model. A 10th order linear prediction (LP), or short‑term, synthesis filter is used which is given by:

, (1)

where  are the (quantified) linear prediction (LP) parameters, and  is the predictor order. The long‑term, or pitch, synthesis filter is given by:

, (2)

where  is the pitch delay and  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 2. 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:

, (3)

where  is the unquantized LP filter and  are the perceptual weighting factors. The values  (for the 12.2 and 10.2 kbit/s mode) or  (for all other modes) and  are used. The weighting filter uses the unquantized LP parameters.

The coder operates on speech frames of 20 ms corresponding to 160 samples at the sampling frequency of 8 000 sample/s. At each 160 speech samples, the speech signal is analysed to extract the parameters of the CELP model (LP filter coefficients, adaptive and fixed codebooks' indices and gains). 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 3. LP analysis is performed twice per frame for the 12.2 kbit/s mode and once for the other modes. For the 12.2 kbit/s mode, the two sets of LP parameters are converted to line spectrum pairs (LSP) and jointly quantized using split matrix quantization (SMQ) with 38 bits. For the other modes, the single set of LP parameters is converted to line spectrum pairs (LSP) and vector quantized using split vector quantization (SVQ). The speech frame is divided into 4 subframes of 5 ms each (40 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 (except for the 5.15 and 4.75 kbit/s modes for which it is done once per frame) based on the perceptually weighted speech signal.

Then the following operations are repeated for each subframe:

The target signal  is computed by filtering the LP residual through the weighted synthesis filter  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,  of the weighted synthesis filter is computed.

Closed‑loop pitch analysis is then performed (to find the pitch lag and gain), using the target  and impulse response , by searching around the open‑loop pitch lag. Fractional pitch with 1/6th or 1/3rd of a sample resolution (depending on the mode) is used.

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

The gains of the adaptive and fixed codebook are scalar quantified with 4 and 5 bits respectively or 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 codec modes is shown in table 1. In each 20 ms speech frame, 95, 103, 118, 134, 148, 159, 204 or 244 bits are produced, corresponding to a bit-rate of 4.75, 5.15, 5.90, 6.70, 7.40, 7.95, 10.2 or 12.2 kbit/s. More detailed bit allocation among the codec parameters is given in tables 9a-9h. Note that the most significant bits (MSB) are always sent first.

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

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Mode | Parameter | 1st subframe | 2nd subframe | 3rd subframe | 4th subframe | total per frame |
|  | 2 LSP sets |  |  |  |  | 38 |
| **12.2 kbit/s** | Pitch delay | 9 | 6 | 9 | 6 | 30 |
| **(GSM EFR)** | Pitch gain | 4 | 4 | 4 | 4 | 16 |
|  | Algebraic code | 35 | 35 | 35 | 35 | 140 |
|  | Codebook gain | 5 | 5 | 5 | 5 | 20 |
|  | **Total** |  |  |  |  | 244 |
|  | LSP set |  |  |  |  | 26 |
| **10.2 kbit/s** | Pitch delay | 8 | 5 | 8 | 5 | 26 |
|  | Algebraic code | 31 | 31 | 31 | 31 | 124 |
|  | Gains | 7 | 7 | 7 | 7 | 28 |
|  | Total |  |  |  |  | 204 |
|  | LSP sets |  |  |  |  | 27 |
| **7.95 kbit/s** | Pitch delay | 8 | 6 | 8 | 6 | 28 |
|  | Pitch gain | 4 | 4 | 4 | 4 | 16 |
|  | Algebraic code | 17 | 17 | 17 | 17 | 68 |
|  | Codebook gain | 5 | 5 | 5 | 5 | 20 |
|  | Total |  |  |  |  | 159 |
|  | LSP set |  |  |  |  | 26 |
| **7.40 kbit/s** | Pitch delay | 8 | 5 | 8 | 5 | 26 |
| **(TDMA EFR)** | Algebraic code | 17 | 17 | 17 | 17 | 68 |
|  | Gains | 7 | 7 | 7 | 7 | 28 |
|  | Total |  |  |  |  | 148 |
|  | LSP set |  |  |  |  | 26 |
| **6.70 kbit/s** | Pitch delay | 8 | 4 | 8 | 4 | 24 |
| **(PDC EFR)** | Algebraic code | 14 | 14 | 14 | 14 | 56 |
|  | Gains | 7 | 7 | 7 | 7 | 28 |
|  | Total |  |  |  |  | 134 |
|  | LSP set |  |  |  |  | 26 |
| **5.90 kbit/s** | Pitch delay | 8 | 4 | 8 | 4 | 24 |
|  | Algebraic code | 11 | 11 | 11 | 11 | 44 |
|  | Gains | 6 | 6 | 6 | 6 | 24 |
|  | Total |  |  |  |  | 118 |
|  | LSP set |  |  |  |  | 23 |
| **5.15 kbit/s** | Pitch delay | 8 | 4 | 4 | 4 | 20 |
|  | Algebraic code | 9 | 9 | 9 | 9 | 36 |
|  | Gains | 6 | 6 | 6 | 6 | 24 |
|  | Total |  |  |  |  | 103 |
|  | LSP set |  |  |  |  | 23 |
| **4.75 kbit/s** | Pitch delay | 8 | 4 | 4 | 4 | 20 |
|  | Algebraic code | 9 | 9 | 9 | 9 | 36 |
|  | Gains | 8 | | 8 | | 16 |
|  | Total |  |  |  |  | 95 |

## 4.4 Principles of the adaptive multi-rate speech decoder

The signal flow at the decoder is shown in figure 4. At the decoder, based on the chosen mode, 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 LSP vectors, the fractional pitch lags, the innovative codevectors, and the pitch and innovative gains. The LSP vectors are converted to the LP filter coefficients and interpolated to obtain LP filters at each subframe. Then, at each 40-sample subframe:

‑ the excitation is constructed by adding the adaptive and innovative codevectors scaled by their respective gains;

‑ the speech is reconstructed by filtering the excitation through the LP synthesis filter.

Finally, the reconstructed speech signal is passed through an adaptive postfilter.

## 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 9a-9h, 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 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 3 are described.

## 5.1 Pre**‑**processing (all modes)

Two pre‑processing functions are applied prior to the encoding process: high‑pass filtering 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 with a cut off frequency of 80 Hz is used, and it is given by:

. (4)

Down‑scaling and high‑pass filtering are combined by dividing the coefficients at the numerator of  by 2.

## 5.2 Linear prediction analysis and quantization

**12.2 kbit/s mode**

Short‑term prediction, or linear prediction (LP), analysis is performed twice per speech frame using the auto‑correlation approach with 30 ms asymmetric windows. No lookahead is used in the auto‑correlation computation.

The auto‑correlations of windowed speech are converted to the LP coefficients using the Levinson‑Durbin algorithm. Then the LP coefficients are transformed to the Line Spectral Pair (LSP) domain for quantization and interpolation purposes. The interpolated quantified and unquantized filter coefficients are converted back to the LP filter coefficients (to construct the synthesis and weighting filters at each subframe).

**10.2, 7.95, 7.40, 6.70, 5.90, 5.15, 4.75 kbit/s modes**

Short‑term prediction, or linear prediction (LP), analysis is performed once per speech frame using the auto‑correlation approach with 30 ms asymmetric windows. A lookahead of 40 samples (5 ms) is used in the auto‑correlation computation.

The auto‑correlations of windowed speech are converted to the LP coefficients using the Levinson‑Durbin algorithm. Then the LP coefficients are transformed to the Line Spectral Pair (LSP) domain for quantization and interpolation purposes. The interpolated quantified and unquantized filter coefficients 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

**12.2 kbit/s mode**

LP analysis is performed twice per frame using two different asymmetric windows. The first window has its weight concentrated at the second subframe and it consists of two halves of Hamming windows with different sizes. The window is given by:

 (5)

The values  and  are used. The second window has its weight concentrated at the fourth subframe and it consists of two parts: the first part is half a Hamming window and the second part is a quarter of a cosine function cycle. The window is given by:

 (6)

where the values  and  are used.

Note that both LP analyses are performed on the same set of speech samples. The windows are applied to 80 samples from past speech frame in addition to the 160 samples of the present speech frame. No samples from future frames are used (no lookahead). A diagram of the two LP analysis windows is depicted below.



Figure 1: LP analysis windows

The auto‑correlations of the windowed speech , are computed by:

 (7)

and a 60 Hz bandwidth expansion is used by lag windowing the auto‑correlations using the window:

, (8)

where  Hz is the bandwidth expansion and  Hz is the sampling frequency. Further,  is multiplied by the white noise correction factor 1.0001 which is equivalent to adding a noise floor at ‑40 dB.

**10.2, 7.95, 7.40, 6.70, 5.90, 5.15, 4.75 kbit/s modes**

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 half a Hamming window and the second part is a quarter of a cosine function cycle. The window is given by equation (6) where the values  and  are used.

The auto‑correlations of the windowed speech , are computed by equation (7) and a 60 Hz bandwidth expansion is used by lag windowing the auto‑correlations using the window of equation (8). Further,  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 (all modes)

The modified auto‑correlations  and  are used to obtain the direct form LP filter coefficients  by solving the set of equations.

 (9)

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



The final solution is given as .

The LP filter coefficients are converted to the line spectral pair (LSP) representation for quantization and interpolation purposes. The conversions to the LSP domain and back to the LP filter coefficient domain are described in the next clause.

### 5.2.3 LP to LSP conversion (all modes)

The LP filter coefficients , are converted to the line spectral pair (LSP) representation for quantization and interpolation purposes. For a 10th order LP filter, the LSPs are defined as the roots of the sum and difference polynomials:

 (10)

and

, (11)

respectively. The polynomial  and  are symmetric and anti‑symmetric, respectively. It can be proven that all roots of these polynomials are on the unit circle and they alternate each other.  has a root  () and  has a root  (). To eliminate these two roots, we define the new polynomials:

 (12)

and

 (13)

Each polynomial has 5 conjugate roots on the unit circle , therefore, the polynomials can be written as

 (14)

and

, (15)

where  with  being the line spectral frequencies (LSF) and they satisfy the ordering property . We refer to  as the LSPs in the cosine domain.

Since both polynomials  and  are symmetric only the first 5 coefficients of each polynomial need to be computed. The coefficients of these polynomials are found by the recursive relations (for  to 4):

 (16)

where  is the predictor order.

The LSPs are found by evaluating the polynomials  and  at 60 points equally spaced between 0 and  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  and . In this method the roots are found directly in the cosine domain . The polynomials  or  evaluated at  can be written as:

,

with:

, (17)

where  is the th order Chebyshev polynomial, and  are the coefficients of either  or , computed using the equations in (16). The polynomial  is evaluated at a certain value of  using the recursive relation:



with initial values  and  The details of the Chebyshev polynomial evaluation method are found in P. Kabal and R.P. Ramachandran [4].

### 5.2.4 LSP to LP conversion (all modes)

Once the LSPs are quantified and interpolated, they are converted back to the LP coefficient domain . The conversion to the LP domain is done as follows. The coefficients of  or  are found by expanding equations (14) and (15) knowing the quantified and interpolated LSPs . The following recursive relation is used to compute :



with initial values  and . The coefficients  are computed similarly by replacing  by .

Once the coefficients  and  are found,  and  are multiplied by  and , respectively, to obtain  and ; that is:

. (18)

Finally the LP coefficients are found by:

. (19)

This is directly derived from the relation , and considering the fact that  and  are symmetric and anti‑symmetric polynomials, respectively.

### 5.2.5 Quantization of the LSP coefficients

**12.2 kbit/s mode**

The two sets of LP filter coefficients per frame are quantified using the LSP representation in the frequency domain; that is:

 (20)

where  are the line spectral frequencies (LSF) in Hz [0,4 000] and  is the sampling frequency. The LSF vector is given by , with *t* denoting transpose.

A 1st order MA prediction is applied, and the two residual LSF vectors are jointly quantified using split matrix quantization (SMQ). The prediction and quantization are performed as follows. Let  and  denote the mean‑removed LSF vectors at frame . The prediction residual vectors  and  are given by:

 (21)

where  is the predicted LSF vector at frame . First order moving‑average (MA) prediction is used where:

, (22)

where  is the quantified second residual vector at the past frame.

The two LSF residual vectors  and  are jointly quantified using split matrix quantization (SMQ). The matrix  is split into 5 submatrices of dimension 2 x 2 (two elements from each vector). For example, the first submatrix consists of the elements , , , and . The 5 submatrices are quantified with 7, 8, 8+1, 8, and 6 bits, respectively. The third submatrix uses a 256‑entry signed codebook (8‑bit index plus 1‑bit sign).

A weighted LSP distortion measure is used in the quantization process. In general, for an input LSP vector  and a quantified vector at index , , the quantization is performed by finding the index  which minimizes:

 (23)

The weighting factors , are given by

 (24)

where  with  and . Here, two sets of weighting coefficients are computed for the two LSF vectors. In the quantization of each submatrix, two weighting coefficients from each set are used with their corresponding LSFs.

**10.2, 7.95, 7.40, 6.70, 5.90, 5.15, 4.75 kbit/s modes**

The set of LP filter coefficients per frame is quantified using the LSP representation in the frequency domain using equation (20).

A 1st order MA prediction is applied, and the residual LSF vector is quantified using split vector quantization. The prediction and quantization are performed as follows. Let  denote the mean‑removed LSF vectors at frame . The prediction residual vectors  is given by:

 (25)

where  is the predicted LSF vector at frame . First order moving‑average (MA) prediction is used where:

, (26)

where  is the quantified residual vector at the past frame and  is the prediction factor for the *j*th LSF.

The LSF residual vectors  is quantified using split vector quantization. The vector  is split into 3 subvectors of dimension 3, 3, and 4. The 3 subvectors are quantified with 7-9 bits according to table 2.

Table 2. Bit allocation split vector quantization of LSF residual vector.

|  |  |  |  |
| --- | --- | --- | --- |
| Mode | Subvector 1 | Subvector 2 | Subvector 3 |
| **10.2 kbit/s** | 8 | 9 | 9 |
| **7.95 kbit/s** | 9 | 9 | 9 |
| **7.40 kbit/s** | 8 | 9 | 9 |
| **6.70 kbit/s** | 8 | 9 | 9 |
| **5.90 kbit/s** | 8 | 9 | 9 |
| **5.15 kbit/s** | 8 | 8 | 7 |
| **4.75 kbit/s** | 8 | 8 | 7 |

The weighted LSP distortion measure of equation (23) with the weighting of equation (24) is used in the quantization process.

### 5.2.6 Interpolation of the LSPs

**12.2 kbit/s mode**

The two sets of quantified (and unquantized) LP parameters are used for the second and fourth subframes whereas the first and third subframes use a linear interpolation of the parameters in the adjacent subframes. The interpolation is performed on the LSPs in the  domain. Let  be the LSP vector at the 4th subframe of the present frame ,  be the LSP vector at the 2nd subframe of the present frame , and  the LSP vector at the 4th subframe of the past frame . The interpolated LSP vectors at the 1st and 3rd subframes are given by:

 (27)

The interpolated LSP vectors are used to compute a different LP filter at each subframe (both quantified and unquantized coefficients) using the LSP to LP conversion method described in clause 5.2.4.

**10.2, 7.95, 7.40, 6.70, 5.90, 5.15, 4.75 kbit/s modes**

The set of quantified (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 subframes. The interpolation is performed on the LSPs in the  domain. The interpolated LSP vectors at the 1st, 2nd, and 3rd subframes are given by:

 (28)

The interpolated LSP vectors are used to compute a different LP filter at each subframe (both quantified and unquantized coefficients) using the LSP to LP conversion method described in clause 5.2.4.

### 5.2.7 Monitoring resonance in the LPC spectrum (all modes)

Resonances in the LPC filter are monitored to detect possible problem areas where divergence between the adaptive codebook memories in the encoder and the decoder could cause unstable filters in areas with highly correlated continuous signals. Typically, this divergence is due to channel errors.

The monitoring of resonance signals is performed using unquantized LSPs . The LSPs are available after the LP to LSP conversion in clause 5.2.3. The algorithm utilises the fact that LSPs are closely located at a peak in the spectrum. First, two distances,  and , are calculated in two different regions, defined as , and .

Either of these two minimum distance conditions must be fulfilled to classify the frame as a resonance frame and increase the resonance counter.



 is a fixed threshold while the second one is depending on  according to:



12 consecutive resonance frames are needed to indicate possible problem conditions, otherwise the LSP\_flag is cleared.



## 5.3 Open‑loop pitch analysis

Open‑loop pitch analysis is performed 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  which is obtained by filtering the input speech signal through the weighting filter . That is, in a subframe of size , the weighted speech is given by:

 (29)

**12.2 kbit/s mode**

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

Open‑loop pitch analysis is performed as follows. In the first step, 3 maxima of the correlation:

 (30)

are found in the three ranges:



The retained maxima , are normalized by dividing by , respectively. The normalized maxima and corresponding delays are denoted by . The winner,  , among the three normalized correlations is selected by favouring the delays with the values in the lower range. This is performed by weighting the normalized correlations corresponding to the longer delays. The best open‑loop delay  is determined as follows:



This procedure of dividing the delay range into 3 clauses and favouring the lower clauses is used to avoid choosing pitch multiples.

**10.2 kbit/s mode**

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:

, (31)

where  is a weighting function. The estimated pitch-lag is the delay that maximises the weighted correlation function . 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, , and a previous frame lag neighbouring emphasis function, :

. (32)

The low pitch lag emphasis function is a given by:

 (33)

where  is defined by a table in the fixed point computational description (ANSI-C code) in [4]. The previous frame lag neighbouring emphasis function depends on the pitch lag of previous speech frames:

 (34)

where ,  is the median filtered pitch lag of 5 previous voiced speech half-frames, and *v* is an adaptive parameter. If the frame is classified as voiced by having the open-loop gain , the *v*-value is set to 1.0 for the next frame. Otherwise, the *v*-value is updated by . The open loop gain is given by:

 (35)

where  is the pitch delay that maximizes . 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*.

**7.95, 7.40, 6.70, 5.90 kbit/s modes**

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

Open‑loop pitch analysis is performed as follows. In the first step, 3 maxima of the correlation in equation (30) are found in the three ranges:



The retained maxima , are normalized by dividing by , respectively. The normalized maxima and corresponding delays are denoted by . The winner,  , among the three normalized correlations is selected by favouring the delays with the values in the lower range. This is performed by weighting the normalized correlations corresponding to the longer delays. The best open‑loop delay  is determined as follows:



This procedure of dividing the delay range into 3 clauses and favouring the lower clauses is used to avoid choosing pitch multiples.

**5.15, 4.75 kbit/s modes**

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

Open‑loop pitch analysis is performed as follows. In the first step, 3 maxima of the correlation in equation (30) are found in the three ranges:



The retained maxima , are normalized by dividing by , respectively. The normalized maxima and corresponding delays are denoted by . The winner,  , among the three normalized correlations is selected by favouring the delays with the values in the lower range. This is performed by weighting the normalized correlations corresponding to the longer delays. The best open‑loop delay  is determined as follows:



This procedure of dividing the delay range into 3 clauses and favouring the lower clauses is used to avoid choosing pitch multiples.

## 5.4 Impulse response computation (all modes)

The impulse response, , of the weighted synthesis filter  is computed each subframe. This impulse response is needed for the search of adaptive and fixed codebooks. The impulse response  is computed by filtering the vector of coefficients of the filter  extended by zeros through the two filters  and .

## 5.5 Target signal computation (all modes)

The target signal for adaptive codebook search is usually computed by subtracting the zero input response of the weighted synthesis filter  from the weighted speech signal . This is performed on a subframe basis.

An equivalent procedure for computing the target signal, which is used in the present document, is the filtering of the LP residual signal  through the combination of synthesis filter  and the weighting filter . 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 clause 5.9.

The residual signal  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 40 as will be explained in the next clause. The LP residual is given by:

 (36)

## 5.6 Adaptive codebook

### 5.6.1 Adaptive codebook search

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.

**12.2 kbit/s mode**

In the first and third subframes, a fractional pitch delay is used with resolutions: 1/6 in the range  and integers only in the range [95, 143]. For the second and fourth subframes, a pitch resolution of 1/6 is always used in the range , where  is nearest integer to the fractional pitch lag of the previous (1st or 3rd) subframe, bounded by 18...143.

Closed‑loop pitch analysis is performed around the open‑loop pitch estimates on a subframe basis. In the first (and third) subframe the range , bounded by 18...143, is searched. For the other subframes, closed‑loop pitch analysis is performed around the integer pitch selected in the previous subframe, as described above. 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.

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:

 (37)

where  is the target signal and  is the past filtered excitation at delay  (past excitation convolved with ). Note that the search range is limited around the open‑loop pitch as explained earlier.

The convolution  is computed for the first delay  in the searched range, and for the other delays in the search range , it is updated using the recursive relation:

,  (38)

and , where , is the excitation buffer. Note that in search stage, the samples, are not known, and they are needed for pitch delays less than 40. To simplify the search, the LP residual is copied to  in order to make the relation in equation (38) valid for all delays.

Once the optimum integer pitch delay is determined, the fractions from –3/6 to 3/6 with a step of 1/6 around that integer are tested. The fractional pitch search is performed by interpolating the normalized correlation in equation (37) and searching for its maximum. The interpolation is performed using an FIR filter  based on a Hamming windowed  function truncated at ± 23 and padded with zeros at ± 24 (). The filter has its cut‑off frequency (‑3 dB) at 3 600 Hz in the over‑sampled domain. The interpolated values of  for the fractions –3/6 to 3/6 are obtained using the interpolation formula:

 (39)

where corresponds to the fractions 0, 1/6, 2/6, 3/6, -2/6, and –1/6, respectively. Note that it is necessary to compute the correlation terms in equation (37) using a range  to allow for the proper interpolation.

Once the fractional pitch lag is determined, the adaptive codebook vector  is computed by interpolating the past excitation signal  at the given integer delay  and phase (fraction) :

 (40)

The interpolation filter  is based on a Hamming windowed  function truncated at ± 59 and padded with zeros at ± 60 (). The filter has a cut‑off frequency (‑3 dB) at 3 600 Hz in the over‑sampled domain.

The adaptive codebook gain is then found by:

 (41)

where  is the filtered adaptive codebook vector (zero state response of  to ).

The computed adaptive codebook gain is quantified using 4‑bit non‑uniform scalar quantization in the range [0.0,1.2].

**7.95 kbit/s mode**

In the first and third subframes, a fractional pitch delay is used with resolutions: 1/3 in the range  and integers only in the range [85, 143]. For the second and fourth subframes, a pitch resolution of 1/3 is always used in the range , where  is nearest integer to the fractional pitch lag of the previous (1st or 3rd) subframe, bounded by 20...143.

Closed‑loop pitch analysis is performed around the open‑loop pitch estimates on a subframe basis. In the first (and third) subframe the range , bounded by 20...143, is searched. For the other subframes, closed‑loop pitch analysis is performed around the integer pitch selected in the previous subframe, as described above. 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 6 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 of equation (37). Note that the search range is limited around the open‑loop pitch as explained earlier.

The convolution  is computed for the first delay  in the searched range, and for the other delays in the search range , it is updated using the recursive relation of equation (38).

Once the optimum integer pitch delay is determined, the fractions from –2/3 to 2/3 with a step of 1/3 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, the adaptive codebook vector  is computed by interpolating the past excitation signal  at the given integer delay and phase (fraction). The interpolation is performed using two FIR filters (Hamming windowed sinc functions); one for interpolating the term in equation (37) with the sinc truncated at ± 11 and the other for interpolating the past excitation with the sinc truncated at ± 29. The filters have their cut‑off frequency (‑3 dB) at 3 600 Hz in the over‑sampled domain.

The adaptive codebook gain is then found as in equation (41).

The computed adaptive codebook gain is quantified using 4‑bit non‑uniform scalar quantization as described in clause 5.8.

**10.2, 7.40 kbit/s mode**

In the first and third subframes, a fractional pitch delay is used with resolutions: 1/3 in the range  and integers only in the range [85, 143]. For the second and fourth subframes, a pitch resolution of 1/3 is always used in the range , where  is nearest integer to the fractional pitch lag of the previous (1st or 3rd) subframe, bounded by 20...143.

Closed‑loop pitch analysis is performed around the open‑loop pitch estimates on a subframe basis. In the first (and third) subframe the range , bounded by 20...143, is searched. For the other subframes, closed‑loop pitch analysis is performed around the integer pitch selected in the previous subframe, as described above. 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.

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 of equation (37). Note that the search range is limited around the open‑loop pitch as explained earlier.

The convolution  is computed for the first delay  in the searched range, and for the other delays in the search range , it is updated using the recursive relation of equation (38).

Once the optimum integer pitch delay is determined, the fractions from –2/3 to 2/3 with a step of 1/3 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, the adaptive codebook vector  is computed by interpolating the past excitation signal  at the given integer delay and phase (fraction). The interpolation is performed using two FIR filters (Hamming windowed sinc functions); one for interpolating the term in equation (37) with the sinc truncated at ± 11 and the other for interpolating the past excitation with the sinc truncated at ± 29. The filters have their cut‑off frequency (‑3 dB) at 3 600 Hz in the over‑sampled domain.

The adaptive codebook gain is then found as in equation (41).

The computed adaptive codebook gain (and the fixed codebook gain) is quantified using 7‑bit non‑uniform vector quantization as described in clause 5.8.

**6.70, 5.90 kbit/s modes**

In the first and third subframes, a fractional pitch delay is used with resolutions: 1/3 in the range  and integers only in the range [85, 143]. For the second and fourth subframes, integer pitch resolution is used in the range , where  is nearest integer to the fractional pitch lag of the previous (1st or 3rd) subframe, bounded by 20...143. Additionally, a fractional resolution of 1/3 is used in the range .

Closed‑loop pitch analysis is performed around the open‑loop pitch estimates on a subframe basis. In the first (and third) subframe the range , bounded by 20...143, is searched. For the other subframes, closed‑loop pitch analysis is performed around the integer pitch selected in the previous subframe, as described above. 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 4 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 of equation (37). Note that the search range is limited around the open‑loop pitch as explained earlier.

The convolution  is computed for the first delay  in the searched range, and for the other delays in the search range , it is updated using the recursive relation of equation (38).

Once the optimum integer pitch delay is determined, the fractions from –2/3 to 2/3 with a step of 1/3 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, the adaptive codebook vector  is computed by interpolating the past excitation signal  at the given integer delay and phase (fraction). The interpolation is performed using two FIR filters (Hamming windowed sinc functions); one for interpolating the term in equation (37) with the sinc truncated at ± 11 and the other for interpolating the past excitation with the sinc truncated at ± 29. The filters have their cut‑off frequency (‑3 dB) at 3 600 Hz in the over‑sampled domain.

The adaptive codebook gain is then found as in equation (41).

The computed adaptive codebook gain (and the fixed codebook gain) is quantified using vector quantization as described in clause 5.8.

**5.15, 4.75 kbit/s modes**

In the first subframe, a fractional pitch delay is used with resolutions: 1/3 in the range  and integers only in the range [85, 143]. For the second, third, and fourth subframes, integer pitch resolution is used in the range , where  is nearest integer to the fractional pitch lag of the previous subframe, bounded by 20...143. Additionally, a fractional resolution of 1/3 is used in the range .

Closed‑loop pitch analysis is performed around the open‑loop pitch estimates on a subframe basis. In the first subframe the range *Top* ± 5, bounded by 20...143, is searched. For the other subframes, closed‑loop pitch analysis is performed around the integer pitch selected in the previous subframe, as described above. The pitch delay is encoded with 8 bits in the first subframe and the relative delay of the other subframes is encoded with 4 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 of equation (37). Note that the search range is limited around the open‑loop pitch as explained earlier.

The convolution  is computed for the first delay  in the searched range, and for the other delays in the search range , it is updated using the recursive relation of equation (38).

Once the optimum integer pitch delay is determined, the fractions from –2/3 to 2/3 with a step of 1/3 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, the adaptive codebook vector  is computed by interpolating the past excitation signal  at the given integer delay and phase (fraction). The interpolation is performed using two FIR filters (Hamming windowed sinc functions); one for interpolating the term in equation (37) with the sinc truncated at ± 11 and the other for interpolating the past excitation with the sinc truncated at ± 29. The filters have their cut‑off frequency (‑3 dB) at 3 600 Hz in the over‑sampled domain.

The adaptive codebook gain is then found as in equation (41).

The computed adaptive codebook gain (and the fixed codebook gain) is quantified using vector quantization as described in clause 5.8.

### 5.6.2 Adaptive codebook gain control (all modes)

The average adaptive codebook gain is calculated if the *LSP\_flag* is set and the unquantized adaptive codebook gain exceeds the gain threshold .

The average gain is calculated from the present unquantized gain and the quantized gains of the seven previous subframes. That is, , where *n* is the current subframe. If the average adaptive codebook gain exceeds the , the unquantized gain is limited to the threshold value and the *GpC\_flag* is set to indicate the limitation.



The *GpC\_flag* is used in the gain quantization in clause 5.8.

## 5.7 Algebraic codebook

### 5.7.1 Algebraic codebook structure

The algebraic codebook structure is based on interleaved single‑pulse permutation (ISPP) design.

**12.2 kbit/s mode**

In this codebook, the innovation vector contains 10 non‑zero pulses. All pulses can have the amplitudes +1 or ‑1. The 40 positions in a subframe are divided into 5 tracks, where each track contains two pulses, as shown in table 3.

Table 3: Potential positions of individual pulses in the algebraic codebook, 12.2 kbit/s.

|  |  |  |
| --- | --- | --- |
| Track | Pulse | Positions |
| 1 | i0, i5 | 0, 5, 10, 15, 20, 25, 30, 35 |
| 2 | i1, i6 | 1, 6, 11, 16, 21, 26, 31, 36 |
| 3 | i2, i7 | 2, 7, 12, 17, 22, 27, 32, 37 |
| 4 | i3, i8 | 3, 8, 13, 18, 23, 28, 33, 38 |
| 5 | i4, i9 | 4, 9, 14, 19, 24, 29, 34, 39 |

Each two pulse positions in one track are encoded with 6 bits (total of 30 bits, 3 bits for the position of every pulse), and the sign of the first pulse in the track is encoded with 1 bit (total of 5 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.

All the 3‑bit pulse positions are Gray coded in order to improve robustness against channel errors. This gives a total of 35 bits for the algebraic code.

**10.2 kbit/s mode**

In this codebook, the innovation vector contains 8 non‑zero pulses. All pulses can have the amplitudes +1 or ‑1. The 40 positions in a subframe are divided into 4 tracks, where each track contains two pulses, as shown in table 4.

Table 4: Potential positions of individual pulses in the algebraic codebook, 10.2 kbit/s.

|  |  |  |
| --- | --- | --- |
| Track | Pulse | Positions |
| 1 | i0, i4 | 0, 4, 8, 12, 16, 20, 24, 28, 32, 36 |
| 2 | i1, i5 | 1, 5, 9, 13, 17, 21, 25, 29, 33, 37 |
| 3 | i2, i6 | 2, 6, 10, 14, 18, 22, 26, 30, 34, 38 |
| 4 | i3, i7 | 3, 7, 11, 15, 19, 23, 27, 31, 35, 39 |

The pulses are grouped into 3, 3, and 2 pulses and their positions are encoded with 10, 10, and 7 bits, respectively (total of 27 bits). The sign of the first pulse in each 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 31 bits for the algebraic code.

**7.95, 7.40 kbit/s modes**

In this codebook, the innovation vector contains 4 non‑zero pulses. All pulses can have the amplitudes +1 or ‑1. The 40 positions in a subframe are divided into 4 tracks, where each track contains one pulse, as shown in table 5.

Table 5: Potential positions of individual pulses in the algebraic codebook, 7.95, 7.40 kbit/s.

|  |  |  |
| --- | --- | --- |
| Track | Pulse | Positions |
| 1 | i0 | 0, 5, 10, 15, 20, 25, 30, 35 |
| 2 | i1 | 1, 6, 11, 16, 21, 26, 31, 36 |
| 3 | i2 | 2, 7, 12, 17, 22, 27, 32, 37 |
| 4 | i3 | 3, 8, 13, 18, 23, 28, 33, 38, |
|  |  | 4, 9, 14, 19, 24, 29, 34, 39 |

The pulse positions are encoded with 3, 3, 3, and 4 bits (total of 13 bits), and the sign of the each pulse is encoded with 1 bit (total of 4 bits). This gives a total of 17 bits for the algebraic code.

**6.70 kbit/s mode**

In this codebook, the innovation vector contains 3 non‑zero pulses. All pulses can have the amplitudes +1 or ‑1. The 40 positions in a subframe are divided into 3 tracks, where each track contains one pulse, as shown in table 6.

Table 6: Potential positions of individual pulses in the algebraic codebook, 6.70 kbit/s.

|  |  |  |
| --- | --- | --- |
| Track | Pulse | Positions |
| 1 | i0 | 0, 5, 10, 15, 20, 25, 30, 35 |
| 2 | i1 | 1, 6, 11, 16, 21, 26, 31, 36, |
|  |  | 3, 8, 13, 18, 23, 28, 33, 38 |
| 3 | i2 | 2, 7, 12, 17, 22, 27, 32, 37, |
|  |  | 4, 9, 14, 19, 24, 29, 34, 39 |

The pulse positions are encoded with 3, 4, and 4 bits (total of 11 bits), and the sign of the each pulse is encoded with 1 bit (total of 3 bits). This gives a total of 14 bits for the algebraic code.

**5.90 kbit/s mode**

In this codebook, the innovation vector contains 2 non‑zero pulses. All pulses can have the amplitudes +1 or ‑1. The 40 positions in a subframe are divided into 2 tracks, where each track contains one pulse, as shown in table 7.

Table 7: Potential positions of individual pulses in the algebraic codebook, 5.90 kbit/s.

|  |  |  |
| --- | --- | --- |
| Track | Pulse | Positions |
| 1 | i0 | 1, 6, 11, 16, 21, 26, 31, 36, |
|  |  | 3, 8, 13, 18, 23, 28, 33, 38 |
| 2 | i1 | 0, 5, 10, 15, 20, 25, 30, 35, |
|  |  | 1, 6, 11, 16, 21, 26, 31, 36, |
|  |  | 2, 7, 12, 17, 22, 27, 32, 37, |
|  |  | 4, 9, 14, 19, 24, 29, 34, 39 |

The pulse positions are encoded with 4 and 5 bits (total of 9 bits), and the sign of the each pulse is encoded with 1 bit (total of 2 bits). This gives a total of 11 bits for the algebraic code.

**5.15, 4.75 kbit/s modes**

In this codebook, the innovation vector contains 2 non‑zero pulses. All pulses can have the amplitudes +1 or ‑1. The 40 positions in a subframe are divided into 5 tracks. Two subsets of 2 tracks each are used for each subframe with one pulse in each track. Different subsets of tracks are used for each subframe. The pulse positions used in each subframe are shown in table 8.

Table 8: Potential positions of individual pulses in the algebraic codebook, 5.15, 4.75 kbit/s.

|  |  |  |  |
| --- | --- | --- | --- |
| Subframe | Subset | Pulse | Positions |
|  | 1 | i0 | 0, 5, 10, 15, 20, 25, 30, 35 |
| **1** |  | i1 | 2, 7, 12, 17, 22, 27, 32, 37 |
|  | 2 | i0 | 1, 6, 11, 16, 21, 26, 31, 36 |
|  |  | i1 | 3, 8, 13, 18, 23, 28, 33, 38 |
|  | 1 | i0 | 0, 5, 10, 15, 20, 25, 30, 35 |
| **2** |  | i1 | 3, 8, 13, 18, 23, 28, 33, 38 |
|  | 2 | i0 | 2, 7, 12, 17, 22, 27, 32, 37 |
|  |  | i1 | 4, 9, 14, 19, 24, 29, 34, 39 |
|  | 1 | i0 | 0, 5, 10, 15, 20, 25, 30, 35 |
| **3** |  | i1 | 2, 7, 12, 17, 22, 27, 32, 37 |
|  | 2 | i0 | 1, 6, 11, 16, 21, 26, 31, 36 |
|  |  | i1 | 4, 9, 14, 19, 24, 29, 34, 39 |
|  | 1 | i0 | 0, 5, 10, 15, 20, 25, 30, 35 |
| **4** |  | i1 | 3, 8, 13, 18, 23, 28, 33, 38 |
|  | 2 | i0 | 1, 6, 11, 16, 21, 26, 31, 36 |
|  |  | i1 | 4, 9, 14, 19, 24, 29, 34, 39 |

One bit is needed to encoded the subset used. The two pulse positions are encoded with 3 bits each (total of 6 bits), and the sign of the each pulse is encoded with 1 bit (total of 2 bits). This gives a total of 9 bits for the algebraic code.

### 5.7.2 Algebraic codebook search

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

 (42)

where  is the filtered adaptive codebook vector and  is the quantified adaptive codebook gain. If  is the algebraic codevector at index , then the algebraic codebook is searched by maximizing the term:

, (43)

where  is the correlation between the target signal  and the impulse response ,  is a the lower triangular Toepliz convolution matrix with diagonal  and lower diagonals , and  is the matrix of correlations of . The vector  (backward filtered target) and the matrix  are computed prior to the codebook search. The elements of the vector  are computed by

, (44)

and the elements of the symmetric matrix  are computed by:

. (45)

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

, (46)

where  is the position of the th pulse,  is its amplitude, and  is the number of pulses ()*.* The energy in the denominator of equation (43) is given by:

 (47)

To simplify the search procedure, the pulse amplitudes are preset by the mere quantization of an appropriate signal . This is simply done by setting the amplitude of a pulse at a certain position equal to the sign of  at that position. The simplification proceeds as follows (prior to the codebook search). First, the sign signal  and the signal  are computed. Second, the matrix  is modified by including the sign information; that is, . The correlation in equation (46) is now given by:

 (48)

and the energy in equation (47) is given by:

 (49)

**12.2 kbit/s mode**

In this case the signal , used for presetting the amplitudes, is a sum of the normalized  vector and normalized long‑term prediction residual :

 (50)

is used. Having preset the pulse amplitudes, as explained above, the optimal pulse positions are determined using an efficient non‑exhaustive analysis‑by‑synthesis search technique. In this technique, the term in equation (43) is tested for a small percentage of position combinations.

First, for each of the five tracks the pulse positions with maximum absolute values of  are searched. From these the global maximum value for all the pulse positions is selected. The first pulse i0 is always set into the position corresponding to the global maximum value.

Next, four iterations are carried out. During each iteration the position of pulse i1 is set to the local maximum of one track. The rest of the pulses are searched in pairs by sequentially searching each of the pulse pairs {i2,i3}, {i4,i5}, {i6,i7} and {i8,i9} in nested loops. Every pulse has 8 possible positions, i.e., there are four 8x8‑loops, resulting in 256 different combinations of pulse positions for each iteration.

In each iteration all the 9 pulse starting positions are cyclically shifted, so that the pulse pairs are changed and the pulse i1 is placed in a local maximum of a different track. The rest of the pulses are searched also for the other positions in the tracks. At least one pulse is located in a position corresponding to the global maximum and one pulse is located in a position corresponding to one of the 4 local maxima.

A special feature incorporated in the codebook is that the selected codevector is filtered through an adaptive pre‑filter  which enhances special spectral components in order to improve the synthesized speech quality. Here the filter  is used, where  is the nearest integer pitch lag to the closed‑loop fractional pitch lag of the subframe, and  is the quantized pitch gain of the current subframe bounded by [0.0,1.0]. Note that prior to the codebook search, the impulse response  must include the pre‑filter . That is, for values of less than 40, the impulse is modified according to

 (50a)

The fixed codebook gain is then found by:

 (51)

where  is the target vector for fixed codebook search and  is the fixed codebook vector convolved with ,

 (52)

**10.2 kbit/s mode**

In this case the signal , used for presetting the amplitudes, is given by eq. (50). Having preset the pulse amplitudes, as explained above, the optimal pulse positions are determined using an efficient non‑exhaustive analysis‑by‑synthesis search technique. In this technique, the term in equation (43) is tested for a small percentage of position combinations.

A special feature incorporated in the codebook is that the selected codevector is filtered through an adaptive pre‑filter  which enhances special spectral components in order to improve the synthesized speech quality. Here the filter  is used, where  is the nearest integer pitch lag to the closed‑loop fractional pitch lag of the subframe, and  is the quantized pitch gain of the previous subframe bounded by [0.0,0.8]. Note that prior to the codebook search, the impulse response  must include the pre‑filter . That is, for values of less than 40, the impulse is modified according to equation (50a).

The fixed codebook gain is then found by equation (51).

**7.95, 7.40 kbit/s modes**

In this case the signal, used for presetting the amplitudes, is equal to the signal . Having preset the pulse amplitudes, as explained above, the optimal pulse positions are determined using an efficient non‑exhaustive analysis‑by‑synthesis search technique. In this technique, the term in equation (43) is tested for a small percentage of position combinations.

A special feature incorporated in the codebook is that the selected codevector is filtered through an adaptive pre‑filter  which enhances special spectral components in order to improve the synthesized speech quality. Here the filter  is used, where  is the nearest integer pitch lag to the closed‑loop fractional pitch lag of the subframe, and  is the quantized pitch gain of the previous subframe bounded by [0.0,0.8]. Note that prior to the codebook search, the impulse response  must include the pre‑filter . That is, for values of less than 40, the impulse is modified according to equation (50a).

The fixed codebook gain is then found by equation (51).

**6.70 kbit/s mode**

In this case the signal , used for presetting the amplitudes, is equal to the signal . Having preset the pulse amplitudes, as explained above, the optimal pulse positions are determined using an efficient non‑exhaustive analysis‑by‑synthesis search technique. In this technique, the term in equation (43) is tested for a small percentage of position combinations.

A special feature incorporated in the codebook is that the selected codevector is filtered through an adaptive pre‑filter  which enhances special spectral components in order to improve the synthesized speech quality. Here the filter  is used, where  is the nearest integer pitch lag to the closed‑loop fractional pitch lag of the subframe, and  is the quantized pitch gain of the previous subframe bounded by [0.0,0.8]. Note that prior to the codebook search, the impulse response  must include the pre‑filter . That is, for values of less than 40, the impulse is modified according to equation (50a).

The fixed codebook gain is then found by equation (51).

**5.90 kbit/s mode**

In this case the signal , used for presetting the amplitudes, is equal to the signal . Having preset the pulse amplitudes, as explained above, the optimal pulse positions are determined using an exhaustive analysis‑by‑synthesis search technique.

A special feature incorporated in the codebook is that the selected codevector is filtered through an adaptive pre‑filter  which enhances special spectral components in order to improve the synthesized speech quality. Here the filter  is used, where  is the nearest integer pitch lag to the closed‑loop fractional pitch lag of the subframe, and  is the quantized pitch gain of the previous subframe bounded by [0.0,0.8]. Note that prior to the codebook search, the impulse response  must include the pre‑filter . That is, for values of less than 40, the impulse is modified according to equation (50a).

The fixed codebook gain is then found by equation (51).

**5.15, 4.75 kbit/s modes**

In this case the signal , used for presetting the amplitudes, is equal to the signal . Having preset the pulse amplitudes, as explained above, the optimal pulse positions are determined using an exhaustive analysis‑by‑synthesis search technique. Note that both subsets are searched.

A special feature incorporated in the codebook is that the selected codevector is filtered through an adaptive pre‑filter  which enhances special spectral components in order to improve the synthesized speech quality. Here the filter  is used, where  is the nearest integer pitch lag to the closed‑loop fractional pitch lag of the subframe, and  is the quantized pitch gain of the previous subframe for the 5.15 kbit/s mode and the previous odd subframe for the 4.75 kbit/s mode bounded by [0.0,0.8]. Note that prior to the codebook search, the impulse response  must include the pre‑filter . That is, for values of less than 40, the impulse is modified according to equation (50a).

The fixed codebook gain is then found by equation (51).

## 5.8 Quantization of the adaptive and fixed codebook gains

### 5.8.1 Adaptive codebook gain limitation in quantization

If the *GpC\_flag* is set, the limited adaptive codebook gain is used in the gain quantization in clause 5.8.2. The quantization codebook search range is limited to only include adaptive codebook gain values less than . This is performed in the quantization search for all modes.

### 5.8.2 Quantization of codebook gains

**Prediction of the fixed codebook gain (all 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  be the mean‑removed innovation energy (in dB) at subframe , and given by:

, (53)

where  is the subframe size,  is the fixed codebook excitation, and  (in dB) is the mean of the innovation energy. The predicted energy is given by:

, (54)

where  are the MA prediction coefficients, and  is the quantified prediction error at subframe . The predicted energy is used to compute a predicted fixed‑codebook gain  as in equation (53) (by substituting  by  and  by ). This is done as follows. First, the mean innovation energy is found by:

 (55)

and then the predicted gain  is found by:

. (56)

A correction factor between the gain  and the estimated one  is given by:

. (57)

Note that the prediction error is given by:

 (58)

**12.2 kbit/s mode**

The correction factor  is computed using a mean energy value,  dB. The correction factor  is quantified using a 5‑bit codebook. The quantization table search is performed by minimizing the error:

. (59)

Once the optimum value  is chosen, the quantified fixed codebook gain is given by .

**10.2 kbit/s mode**

The correction factor  is computed using a mean energy value,  dB. The adaptive codebook gain and the correction factor  are jointly vector quantized using a 7-bit codebook. The gain codebook search is performed by minimizing equation (63).

**7.95 kbit/s mode**

The correction factor  is computed using a mean energy value,  dB. The same scalar codebooks as for the 12.2 kbit/s mode is used for quantization of the adaptive codebook gain  and the correction factor . The search of the codebooks starts with finding 3 candidates for the adaptive codebook gain. These candidates are the best codebook value in scalar quantization and the two adjacent codebook values. These 3 candidates are searched together with the correction factor codebook minimizing the term of equation (63).

An adaptor based on the coding gain in the adaptive codebook decides if the coding gain is low. If this is the case, the correction factor codebook is searched once more minimizing a modified criterion in order to find a new quantized fixed codebook gain. The modified criterion is given by:

 (60)

where  and  are the energy (the squared norm) of the LP residual and the total excitation, respectively. The criterion is searched with the already quantized adaptive codebook gain and the correction factor  that minimizes (60) is selected. The balance factor  decides the amount of energy matching in the modified criterion. This factor is adaptively decided based on the coding gain in the adaptive codebook as computed by:

. (61)

If the coding gain *ag* is less than 1 dB, the modified criterion is employed, except when an onset is detected. An onset is said to be detected if the fixed codebook gain in the current subframe is more than twice the value of the fixed codebook gain in the previous subframe. A hangover of 8 subframes is used in the onset detection so that the modified criterion is not used for the next 7 subframes either if an onset is detected. The balance factor  is computed from the median filtered adaptive coding gain. The current and the *ag*-values for the previous 4 subframes are median filtered to get . The -factor is computed by:

. (62)

**7.40 kbit/s mode**

The correction factor  is computed using a mean energy value,  dB. The adaptive codebook gain and the correction factor  are jointly vector quantized using a 7-bit codebook. The gain codebook search is performed by minimizing the square of the weighted error between original and reconstructed speech which is given by

 (63)

where  is the target vector,  is the filtered adaptive codebook vector, and  is the filtered fixed codebook vector.

**6.70 kbit/s mode**

The correction factor  is computed using a mean energy value,  dB. The adaptive codebook gain and the correction factor  are jointly vector quantized using a 7-bit codebook. The gain codebook search is performed by minimizing equation (63).

**5.90, 5.15 kbit/s modes**

The correction factor  is computed using a mean energy value,  dB. The adaptive codebook gain and the correction factor  are jointly vector quantized using a 6-bit codebook. The gain codebook search is performed by minimizing equation (63).

**4.75 kbit/s mode**

The correction factors are computed using a mean energy value,  dB. The adaptive codebook gains and the correction factors are jointly vector quantized every 10 ms. This is done by minimizing a weighted sum of the error criterion (63) for each of the two subframes. The default values on the weighing factors are 1. If the energy of the second subframe is more than two times the energy of the first subframe, the weight of the first subframe is set to 2. If the energy of the first subframe is more than four times the energy of the second subframe, the weight of the second subframe is set to 2.

### 5.8.3 Update past quantized adaptive codebook gain buffer (all modes)

After the gain quantization, the buffer with past adaptive codebook gains is updated, regardless of the value of the *GpC\_flag.* That is, .

## 5.9 Memory update (all modes)

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 are quantified, the excitation signal, , in the present subframe is found by:

, (64)

where  and  are the quantified adaptive and fixed codebook gains, respectively,  the adaptive codebook vector (interpolated past excitation), and  is the fixed codebook vector (algebraic code including pitch sharpening). The states of the filters can be updated by filtering the signal  (difference between residual and excitation) through the filters  and  for the 40‑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 synthesized speech, , is computed by filtering the excitation signal through . The output of the filter due to the input  is equivalent to . So the states of the synthesis filter  are given by . Updating the states of the filter  can be done by filtering the error signal  through this filter to find the perceptually weighted error . However, the signal  can be equivalently found by:

, (65)

Since the signals , , and  are available, the states of the weighting filter are updated by computing  as in equation (65) for . This saves two filterings.

**4.75 kbit/s mode**

The memory update in the first and third subframes use the unquantized gains in equation (64). After the second and fourth subframes respectively, when the gains are quantized, the state is recalculated using the quantized gains.

# 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 performing synthesis to obtain the reconstructed speech. The reconstructed speech is then post‑filtered and upscaled. The signal flow at the decoder is shown in figure 4.

## 6.1 Decoding and speech synthesis

The decoding process is performed in the following order:

**Decoding of LP filter parameters:** The received indices of LSP quantization are used to reconstruct the quantified LSP vectors. The interpolation described in clause 5.2.6 is performed to obtain 4 interpolated LSP vectors (corresponding to 4 subframes). For each subframe, the interpolated LSP vector is converted to LP filter coefficient domain , 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  is found by interpolating the past excitation  (at the pitch delay) using the FIR filter described in clause 5.6.

2) **Decoding of the innovative codebook 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 . If the integer part of the pitch lag, *T*, is less than the subframe size 40, the pitch sharpening procedure is applied which translates into modifying  by , where  is the decoded pitch gain, , bounded by [0.0,1.0] or [0.0,0.8], depending on mode.

3) **Decoding of the adaptive and fixed codebook gains:** In case of scalar quantization of the gains (12.2 kbit/s and 7.95 kbit/s modes) the received indices are used to readily find the quantified adaptive codebook gain, , and the quantified fixed codebook gain correction factor, , from the corresponding quantization tables. In case of vector quantization of the gains (all other modes), the received index gives both the quantified adaptive codebook gain, , and the quantified fixed codebook gain correction factor, . The estimated fixed codebook gain  is found as described in clause 5.7. First, the predicted energy is found by:

 (66)

and then the mean innovation energy is found by:

. (67)

The predicted gain  is found by:

. (68)

The quantified fixed codebook gain is given by:

. (69)

4) **Smoothing of the fixed codebook gain (10.2, 6.70, 5.90, 5.15, 4.75 kbit/s modes):** An adaptive smoothing of the fixed codebook gain is performed to avoid unnatural fluctuations in the energy contour. The smoothing is based on a measure of the stationarity of the short-term spectrum in the **q** domain. The smoothing strength is computed from this measure. An averaged **q**-value is computed for each frame *n* by:

. (70)

For each subframe *m*, a difference measure between the averaged vector and the quantized and interpolated vector is computed by:

, (71)

where *j* runs over the 10 LSPs. Furthermore, a smoothing factor,, is computed by:

, (72)

where the constants are set to  and . A hangover period of 40 subframes is used where the -value is set 1.0 if the  has been above 0.65 for 10 consecutive frames. A value of 1.0 corresponds to no smoothing. An averaged fixed codebook gain value is computed for each subframe by:

. (73)

The fixed codebook gain used for synthesis is now replaced by a smoothed value given by:

. (74)

5) **Anti-sparseness processing (7.95, 6.70, 5.90, 5.15, 4.75 kbit/s modes):** An adaptive anti-sparseness post-processing procedure is applied to the fixed codebook vector in order to reduce perceptual artefacts 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  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:



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

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

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

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

6) **Computing the reconstructed speech:** The excitation at the input of the synthesis filter is given by:

. (75)

Before the speech synthesis, a post‑processing of excitation elements is performed. This means that the total excitation is modified by emphasizing the contribution of the adaptive codebook vector:

 (76)

Adaptive gain control (AGC) is used to compensate for the gain difference between the non‑emphasized excitation  and emphasized excitation  The gain scaling factor *η* for the emphasized excitation is computed by:

 (77)

The gain‑scaled emphasized excitation signal  is given by:

. (78)

The reconstructed speech for the subframe of size 40 is given by:

. (79)

where  are the interpolated LP filter coefficients.

7) **Additional instability protection**: An additional instability protection is implemented in the speech decoder which is monitoring overflows in the synthesis filter. If an overflow has occurred in the synthesis part, the whole adaptive codebook memory,  is scaled down by a factor of 4, and the synthesis filtering is repeated using this down-scaled memory. I.e. in this case step 6) is repeated, except that the post-processing in (76) - (78) of the excitation signal is by-passed.

The synthesized speech  is then passed through an adaptive postfilter which is described in the following clause.

## 6.2 Post‑processing

### 6.2.1 Adaptive post‑filtering (all modes)

The adaptive postfilter is the cascade of two filters: a formant postfilter, and a tilt compensation filter. The postfilter is updated every subframe of 5 ms.

The formant postfilter is given by:

 (80)

where  is the received quantified (and interpolated) LP inverse filter (LP analysis is not performed at the decoder), and the factors  and  control the amount of the formant post‑filtering.

Finally, the filter  compensates for the tilt in the formant postfilter  and is given by:

 (81)

where  is a tilt factor, with  being the first reflection coefficient calculated on the truncated () impulse response, , of the filter .  is given by:

. (82)

The post‑filtering process is performed as follows. First, the synthesized speech  is inverse filtered through  to produce the residual signal . The signal  is filtered by the synthesis filter . Finally, the signal at the output of the synthesis filter  is passed to the tilt compensation filter  resulting in the post‑filtered speech signal .

Adaptive gain control (AGC) is used to compensate for the gain difference between the synthesized speech signal  and the post‑filtered signal . The gain scaling factor  for the present subframe is computed by:

. (83)

The gain‑scaled post‑filtered signal  is given by:

 (84)

where  is updated in sample‑by‑sample basis and given by:

 (85)

where  is a AGC factor with value of 0.9.

**12.2, 10.2 kbit/s modes**

The adaptive post‑filtering factors are given by: ,  and

. (86)

**7.95, 7.40, 6.70, 5.90, 5.15, 4.75 kbit/s modes**

The adaptive post‑filtering factors are given by: ,  and .

### 6.2.2 High-pass filtering and up-scaling (all modes)

The high-pass filter serves as a precaution against undesired low frequency components. A filter cut-off frequency of 60 Hz is used, and the filter is given by

. (87)

Up‑scaling consists of multiplying the post‑filtered speech by a factor of 2 to compensate for the down‑scaling by 2 which is applied to the input signal.

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

The detailed allocation of the bits in the adaptive multi-rate speech encoder is shown for each mode in table 9a-9h. 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 9a: Source encoder output parameters in order of occurrence and bit allocation within the speech frame of 244 bits/20 ms, 12.2 kbit/s mode.

|  |  |
| --- | --- |
| Bits (MSB‑LSB) | Description |
| s1 ‑ s7 | index of 1st LSF submatrix |
| s8 ‑ s15 | index of 2nd LSF submatrix |
| s16 ‑ s23 | index of 3rd LSF submatrix |
| s24 | sign of 3rd LSF submatrix |
| s25 ‑ s32 | index of 4th LSF submatrix |
| s33 ‑ s38 | index of 5th LSF submatrix |
| subframe 1 | |
| s39 ‑ s47 | adaptive codebook index |
| s48 ‑ s51 | adaptive codebook gain |
| s52 | sign information for 1st and 6th pulses |
| s53 ‑ s55 | position of 1st pulse |
| s56 | sign information for 2nd and 7th pulses |
| s57 ‑ s59 | position of 2nd pulse |
| s60 | sign information for 3rd and 8th pulses |
| s61 ‑ s63 | position of 3rd pulse |
| s64 | sign information for 4th and 9th pulses |
| s65 ‑ s67 | position of 4th pulse |
| s68 | sign information for 5th and 10th pulses |
| s69 ‑ s71 | position of 5th pulse |
| s72 ‑ s74 | position of 6th pulse |
| s75 ‑ s77 | position of 7th pulse |
| s78 ‑ s80 | position of 8th pulse |
| s81 ‑ s83 | position of 9th pulse |
| s84 ‑ s86 | position of 10th pulse |
| s87 ‑ s91 | fixed codebook gain |
| subframe 2 | |
| s92 ‑ s97 | adaptive codebook index (relative) |
| s98 ‑ s141 | same description as s48 ‑ s91 |
| subframe 3 | |
| s142 ‑ s194 | same description as s39 ‑ s91 |
| subframe 4 | |
| s195 ‑ s244 | same description as s92 ‑ s141 |

Table 9b: Source encoder output parameters in order of occurrence and bit allocation within the speech frame of 204 bits/20 ms, 10.2 kbit/s mode.

|  |  |
| --- | --- |
| Bits (MSB‑LSB) | Description |
| s1 – s8 | index of 1st LSF subvector |
| s9 ‑ s17 | index of 2nd LSF subvector |
| s18 – s26 | index of 3rd LSF subvector |
| subframe 1 | |
| s27 – s34 | adaptive codebook index |
| s35 | sign information for 1st and 5th pulses |
| s36 | sign information for 2nd and 6th pulses |
| s37 | sign information for 3rd and 7th pulses |
| s38 | sign information for 4th and 8th pulses |
| s39-s48 | position for 1st, 2nd, and 5th pulses |
| s49-s58 | position for 3rd, 6th, and 7th pulses |
| s59-s65 | position for 4th and 8th pulses |
| s66 – s72 | codebook gains |
| subframe 2 | |
| s73 – s77 | adaptive codebook index (relative) |
| s78 – s115 | same description as s35 – s72 |
| subframe 3 | |
| s116 – s161 | same description as s27 – s72 |
| subframe 4 | |
| s162 – s204 | same description as s73 – s115 |

Table 9c: Source encoder output parameters in order of occurrence and bit allocation within the speech frame of 159 bits/20 ms, 7.95 kbit/s mode.

|  |  |
| --- | --- |
| Bits (MSB‑LSB) | Description |
| s1 – s9 | index of 1st LSF subvector |
| s10 ‑ s18 | index of 2nd LSF subvector |
| s19 – s27 | index of 3rd LSF subvector |
| subframe 1 | |
| s28 – s35 | adaptive codebook index |
| s36 – s39 | position of 4th pulse |
| s40 – s42 | position of 3rd pulse |
| s43 – s45 | position of 2nd pulse |
| s46 – s48 | position of 1st pulse |
| s49 | sign information for 4th pulse |
| s50 | sign information for 3rd pulse |
| s51 | sign information for 2nd pulse |
| s52 | sign information for 1st pulse |
| s53 – s56 | adaptive codebook gain |
| s57 – s61 | fixed codebook gain |
| subframe 2 | |
| s62 – s67 | adaptive codebook index (relative) |
| s68 – s93 | same description as s36 – s61 |
| subframe 3 | |
| s94 – s127 | same description as s28 – s61 |
| subframe 4 | |
| s128 – s159 | same description as s62 – s93 |

Table 9d: Source encoder output parameters in order of occurrence and bit allocation within the speech frame of 148 bits/20 ms, 7.40 kbit/s mode.

|  |  |
| --- | --- |
| Bits (MSB‑LSB) | Description |
| s1 – s8 | index of 1st LSF subvector |
| s9 ‑ s17 | index of 2nd LSF subvector |
| s18 – s26 | index of 3rd LSF subvector |
| subframe 1 | |
| s27 – s34 | adaptive codebook index |
| s35 – s38 | position of 4th pulse |
| s39 – s41 | position of 3rd pulse |
| s42 - s44 | position of 2nd pulse |
| s45 – s47 | position of 1st pulse |
| s48 | sign information for 4th pulse |
| s49 | sign information for 3rd pulse |
| s50 | sign information for 2ndd pulse |
| s51 | sign information for 1st pulse |
| s52 – s58 | codebook gains |
| subframe 2 | |
| s59 – s63 | adaptive codebook index (relative) |
| s64 – s87 | same description as s35 – s58 |
| subframe 3 | |
| s88 – s119 | same description as s27 – s58 |
| subframe 4 | |
| s120 – s148 | same description as s59 – s87 |

Table 9e: Source encoder output parameters in order of occurrence and bit allocation within the speech frame of 134 bits/20 ms, 6.70 kbit/s mode.

|  |  |
| --- | --- |
| Bits (MSB‑LSB) | Description |
| s1 – s8 | index of 1st LSF subvector |
| s9 ‑ s17 | index of 2nd LSF subvector |
| s18 – s26 | index of 3rd LSF subvector |
| subframe 1 | |
| s27 – s34 | adaptive codebook index |
| s35 – s38 | position of 3rd pulse |
| s39 – s42 | position of 2nd pulse |
| s43 – s45 | position of 1st pulse |
| s46 | sign information for 3rd pulse |
| s47 | sign information for 2nd pulse |
| s48 | sign information for 1st pulse |
| s49 – s55 | codebook gains |
| subframe 2 | |
| s56 – s59 | adaptive codebook index (relative) |
| s60 – s80 | same description as s35 – s55 |
| subframe 3 | |
| s81 – s109 | same description as s27 – s55 |
| subframe 4 | |
| s110 – s134 | same description as s56 – s80 |

Table 9f: Source encoder output parameters in order of occurrence and bit allocation within the speech frame of 118 bits/20 ms, 5.90 kbit/s mode.

|  |  |
| --- | --- |
| Bits (MSB‑LSB) | Description |
| s1 – s8 | index of 1st LSF subvector |
| s9 ‑ s17 | index of 2nd LSF subvector |
| s18 – s26 | index of 3rd LSF subvector |
| subframe 1 | |
| s27 – s34 | adaptive codebook index |
| s35 – s39 | position of 2nd pulse |
| s40 – s43 | position of 1st pulse |
| s44 | sign information for 2nd pulse |
| s45 | sign information for 1st pulse |
| s46 – s51 | codebook gains |
| subframe 2 | |
| s52 – s55 | adaptive codebook index (relative) |
| s56 – s72 | same description as s35 – s51 |
| subframe 3 | |
| s73 – s97 | same description as s27 – s51 |
| subframe 4 | |
| s98 – s118 | same description as s52 – s72 |

Table 9g: Source encoder output parameters in order of occurrence and bit allocation within the speech frame of 103 bits/20 ms, 5.15 kbit/s mode.

|  |  |
| --- | --- |
| Bits (MSB‑LSB) | Description |
| s1 – s8 | index of 1st LSF subvector |
| s9 ‑ s16 | index of 2nd LSF subvector |
| s17 – s23 | index of 3rd LSF subvector |
| subframe 1 | |
| s24 – s31 | adaptive codebook index |
| s32 | position subset |
| s33 – s35 | position of 2nd pulse |
| s36 – s38 | position of 1st pulse |
| s39 | sign information for 2nd pulse |
| s40 | sign information for 1st pulse |
| s41 – s46 | codebook gains |
| subframe 2 | |
| s47 – s50 | adaptive codebook index (relative) |
| s51 – s65 | same description as s32 – s46 |
| subframe 3 | |
| s66 – s84 | same description as s47 – s65 |
| subframe 4 | |
| s85 – s103 | same description as s47 – s65 |

Table 9h: Source encoder output parameters in order of occurrence and bit allocation within the speech frame of 95 bits/20 ms, 4.75 kbit/s mode.

|  |  |
| --- | --- |
| Bits (MSB‑LSB) | Description |
| s1 – s8 | index of 1st LSF subvector |
| s9 ‑ s16 | index of 2nd LSF subvector |
| s17 – s23 | index of 3rd LSF subvector |
| subframe 1 | |
| s24 – s31 | adaptive codebook index |
| s32 | position subset |
| s33 – s35 | position of 2nd pulse |
| s36 – s38 | position of 1st pulse |
| s39 | sign information for 2nd pulse |
| s40 | sign information for 1st pulse |
| s41 – s48 | codebook gains |
| subframe 2 | |
| s49 – s52 | adaptive codebook index (relative) |
| s53 – s61 | same description as s32 – s40 |
| subframe 3 | |
| s62 - s65 | same description as s49 – s52 |
| s66 – s82 | same description as s32– s48 |
| subframe 4 | |
| s83 – s95 | same description as s49 – s61 |

# 8 Homing sequences

## 8.1 Functional description

The adaptive multi-rate speech codec is described in a bit‑exact arithmetic to allow for easy type approval as well as general testing purposes of the adaptive multi-rate speech codec.

The response of the codec to a predefined input sequence can only be foreseen if the internal state variables of the codec are in a predefined state at the beginning of the experiment. Therefore, the codec has to be put in a so called home state before a bit‑exact test can be performed. This is usually done by a reset (a procedure in which the internal state variables of the codec are set to their defined initial values). The codec mode of the speech encoder and speech decoder shall be set to the tested codec mode by external means at reset.

To allow a reset of the codec in remote locations, special homing frames have been defined for the encoder and the decoder, thus enabling a codec homing by inband signalling.

The codec homing procedure is defined in such a way, that in either direction (encoder or decoder) the homing functions are called after processing the homing frame that is input. The output corresponding to the first homing frame is therefore dependent on the used codec mode and the codec state when receiving that frame and hence usually not known. The response of the encoder to any further homing frame is by definition the corresponding decoder homing frame for the used codec mode. The response of the decoder to any further homing frame is by definition the encoder homing frame. This procedure allows homing of both, the encoder and decoder from either side, if a loop back configuration is implemented, taking proper framing into account.

## 8.2 Definitions

**Encoder homing frame:** The encoder homing frame consists of 160 identical samples, each 13 bits long, with the least significant bit set to "one" and all other bits set to "zero". When written to 16‑bit words with left justification, the samples have a value of 0008 hex. The speech decoder has to produce this frame as a response to the second and any further decoder homing frame if at least two decoder homing frames were input to the decoder consecutively. The encoder homing frame is identical for all codec modes.

**Decoder homing frame:** There exist eight different decoder homing frames, which correspond to the eight AMR codec modes. Using one of these codec modes, the corresponding decoder homing frame is the natural response of the speech encoder to the second and any further encoder homing frame if at least two encoder homing frames were input to the encoder consecutively. In [4], for each decoder homing frame the parameter values are given.

## 8.3 Encoder homing

Whenever the adaptive multi-rate speech encoder receives at its input an encoder homing frame exactly aligned with its internal speech frame segmentation, the following events take place:

Step 1: The speech encoder performs its normal operation including VAD and SCR and produces in accordance with the used codec mode a speech parameter frame at its output which is in general unknown. But if the speech encoder was in its home state at the beginning of that frame, then the resulting speech parameter frame is identical to that decoder homing frame, which corresponds to the used codec mode (this is the way how the decoder homing frames were constructed).

Step 2: After successful termination of that operation the speech encoder provokes the homing functions for all sub‑modules including VAD and SCR and sets all state variables into their home state. On the reception of the next input frame, the speech encoder will start from its home state.

NOTE: Applying a sequence of N encoder homing frames will cause at least N‑1 decoder homing frames at the output of the speech encoder.

## 8.4 Decoder homing

Whenever the speech decoder receives at its input a decoder homing frame, which corresponds to the used codec mode, then the following events take place:

Step 1: The speech decoder performs its normal operation and produces a speech frame at its output which is in general unknown. But if the speech decoder was in its home state at the beginning of that frame, then the resulting speech frame is replaced by the encoder homing frame. This would not naturally be the case but is forced by this definition here.

Step 2: After successful termination of that operation the speech decoder provokes the homing functions for all sub‑modules including the comfort noise generator and sets all state variables into their home state. On the reception of the next input frame, the speech decoder will start from its home state.

NOTE 1: Applying a sequence of N decoder homing frames will cause at least N‑1 encoder homing frames at the output of the speech decoder.

NOTE 2: By definition (!) the first frame of each decoder test sequence must differ from the decoder homing frame at least in one bit position within the parameters for LPC and first subframe. Therefore, if the decoder is in its home state, it is sufficient to check only these parameters to detect a subsequent decoder homing frame. This definition is made to support a delay‑optimized implementation in the TRAU uplink direction.



Figure 2: Simplified block diagram of the CELP synthesis model



Figure 3: Simplified block diagram of the adaptive multi-rate encoder



Figure 4: Simplified block diagram of the adaptive multi-rate decoder

# 9 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 artefacts in CELP", in *Proc. ICASSP’98*, pp. I-145‑I-148.

Annex A (informative):  
Change history

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Tdoc** | | **SPEC** | | **CR** | **REV** | | **VERS** | | **SUBJECT** | | | **CAT** | | **NEW\_VERS** |
| SP-99570 | | 26.090 | | A001 |  | | 3.0.1 | | Bit allocation of the adaptive multi-rate codec | | | F | | 3.1.0 |
| **Change history** | | | | | | | | | | | | | | |
| **Date** | **TSG #** | | **TSG Doc.** | | | **CR** | | **Rev** | | **Subject/Comment** | **Old** | | **New** | |
| 03-2001 | 11 | |  | | |  | |  | | Version for Release 4 |  | | 4.0.0 | |
| 06-2002 | 16 | |  | | |  | |  | | Version for Release 5 | 4.0.0 | | 5.0.0 | |
| 12-2004 | 26 | |  | | |  | |  | | Version for Release 6 | 5.0.0 | | 6.0.0 | |
| 06-2007 | 36 | |  | | |  | |  | | Version for Release 7 | 6.0.0 | | 7.0.0 | |
| 12-2008 | 42 | |  | | |  | |  | | Version for Release 8 | 7.0.0 | | 8.0.0 | |
| 06-2009 | 44 | | SP-090249 | | | 0002 | | 1 | | Corrections to Quantization of codebook gains in sub-clause 5.8.2 | 8.0.0 | | 8.1.0 | |
| 06-2009 | 44 | | SP-090249 | | | 0003 | | 1 | | Correction to recursive equation for the past filtered excitation in sub-clause 5.6.1 | 8.0.0 | | 8.1.0 | |
| 12-2009 | 46 | |  | | |  | |  | | Version for Release 9 | 8.1.0 | | 9.0.0 | |
| 03-2011 | 51 | |  | | |  | |  | | Version for Release 10 | 9.0.0 | | 10.0.0 | |
| 09-2011 | 53 | | SP-110548 | | | 0007 | | 1 | | Correction of equation for pitch sharpening | 10.0.0 | | 10.1.0 | |
| 09-2012 | 57 | |  | | |  | |  | | Version for Release 11 | 10.1.0 | | 11.0.0 | |
| 09-2014 | 65 | |  | | |  | |  | | Version for Release 12 | 11.0.0 | | 12.0.0 | |
| 12-2015 | 70 | |  | | |  | |  | | Version for Release 13 | 12.0.0 | | 13.0.0 | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Change history** | | | | | | | |
| **Date** | **Meeting** | **TDoc** | **CR** | **Rev** | **Cat** | **Subject/Comment** | **New version** |
| 2017-03 | 75 |  |  |  |  | Version for Release 14 | 14.0.0 |
| 2018-06 | 80 |  |  |  |  | Version for Release 15 | 15.0.0 |
| 2020-07 | - | - | - | - | - | Update to Rel-16 version (MCC) | 16.0.0 |
| 2022-04 | - | - | - | - | - | Update to Rel-17 version (MCC) | 17.0.0 |
| 2024-03 | - | - | - | - | - | Update to Rel-18 version (MCC) | 18.0.0 |