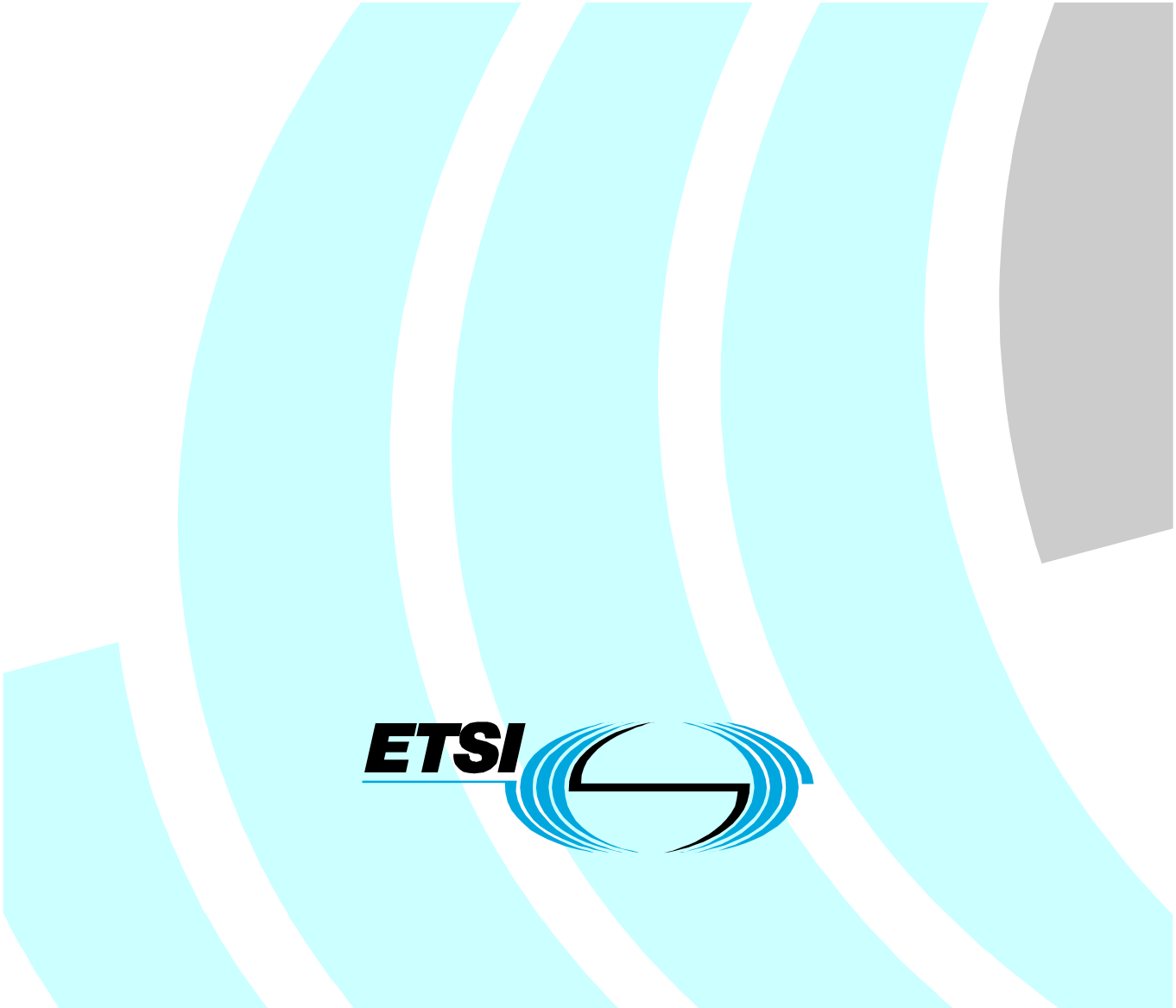
Final draft ETSI ES 202 211 V1.1.1 (2003-08)

ETSI Standard

Speech Processing, Transmission and Quality Aspects (STQ); Distributed speech recognition; Extended front-end feature extraction algorithm; Compression algorithms; Back-end speech reconstruction algorithm



Reference
DES/STQ-00030

Keywords
performance, speech, transmission

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Important notice

Individual copies of the present document can be downloaded from:
http://www.etsi.org

The present document may be made available in more than one electronic version or in print. In any case of existing or perceived difference in contents between such versions, the reference version is the Portable Document Format (PDF). In case of dispute, the reference shall be the printing on ETSI printers of the PDF version kept on a specific network drive within ETSI Secretariat.

Users of the present document should be aware that the document may be subject to revision or change of status. Information on the current status of this and other ETSI documents is available at
http://portal.etsi.org/tb/status/status.asp

If you find errors in the present document, send your comment to:
editor@etsi.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.

© European Telecommunications Standards Institute 2003.
All rights reserved.

DECT™, **PLUGTESTS**™ and **UMTS**™ are Trade Marks of ETSI registered for the benefit of its Members.
TIPHON™ and the **TIPHON logo** are Trade Marks currently being registered by ETSI 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.

Contents

Intellectual Property Rights 5
Foreword 5
Introduction 5
1 Scope 6
2 References 7
3 Definitions, symbols and abbreviations 7
3.1 Definitions 7
3.2 Symbols 8
3.3 Abbreviations 9
4 Front-end feature extraction algorithm 10
4.1 Introduction 10
4.2 Front-end algorithm description 10
4.2.1 Front-end block diagram 10
4.2.2 Analog-to-digital conversion 11
4.2.3 Offset compensation 11
4.2.4 Framing 11
4.2.5 Energy measure 11
4.2.6 Pre-Emphasis (PE) 11
4.2.7 Windowing (W) 12
4.2.8 Fast Fourier Transform (FFT) 12
4.2.9 Mel-Filtering (MF) 12
4.2.10 Non-linear transformation 13
4.2.11 Cepstral coefficients 13
4.2.12 Voice Activity Detection (VAD) 13
4.2.13 Low-Band Noise Detection (LBND) 18
4.2.14 Pre-Processing for pitch and class estimation 19
4.2.15 Pitch estimation 20
4.2.15.1 Dirichlet interpolation 20
4.2.15.2 Non-speech and low-energy frames 22
4.2.15.3 Search ranges specification and processing 22
4.2.15.4 Spectral peaks determination 22
4.2.15.5 F0 Candidates generation 24
4.2.15.6 Computing correlation scores 26
4.2.15.7 Pitch estimate selection 28
4.2.15.8 History information update 30
4.2.15.9 Output pitch value 31
4.2.16 Classification 31
4.2.17 Front-end output 32
5 Feature compression algorithm 32
5.1 Introduction 32
5.2 Compression algorithm description 32
5.2.1 Input 32
5.2.2 Vector quantization 33
5.2.3 Pitch and class quantization 33
5.2.3.1 Class quantization 34
5.2.3.2 Pitch quantization 34
6 Framing, bit-stream formatting, and error protection 35
6.1 Introduction 35
6.2 Algorithm description 36
6.2.1 Multiframe format 36
6.2.2 Synchronization sequence 36
6.2.3 Header field 36

6.2.4 Frame Packet Stream 38
7 Bit-stream decoding and error mitigation 38
7.1 Introduction 38
7.2 Algorithm description 38
7.2.1 Synchronization sequence detection 38
7.2.2 Header decoding 39
7.2.3 Feature decompression 39
7.2.4 Error mitigation 39
7.2.4.1 Detection of frames received with errors 39
7.2.4.2 Substitution of parameter values for frames received with errors 40
7.2.4.3 Modification of parameter values for frames received with errors 40
8 Server side speech reconstruction 43
8.1 Introduction 43
8.2 Algorithm description 43
8.2.1 Speech reconstruction block diagram 43
8.2.2 Pitch tracking and smoothing 43
8.2.2.1 First stage - gross pitch error correction 44
8.2.2.2 Second stage - voiced/unvoiced decision and other corrections 46
8.2.2.3 Third stage - smoothing 47
8.2.2.4 Voicing class correction 47
8.2.3 Harmonic structure initialization 48
8.2.4 Unvoiced Phase (UPH) synthesis 48
8.2.5 Harmonic magnitudes reconstruction 48
8.2.5.1 High order cepstra recovery 49
8.2.5.2 Solving front-end equation 57
8.2.5.3 Cepstra to magnitudes transformation 61
8.2.5.4 Combined magnitudes estimate calculation 63
8.2.5.4.1 Combined magnitude estimate for unvoiced harmonics 63
8.2.5.4.2 Combined magnitude estimate for voiced harmonics 64
8.2.6 All-pole spectral envelope modelling 65
8.2.7 Postfiltering 67
8.2.8 Voiced phase synthesis 68
8.2.9 Line spectrum to time-domain transformation 70
8.2.9.1 Mixed-voiced frames processing 70
8.2.9.2 Filtering very high-frequency harmonics 70
8.2.9.3 Energy normalization 71
8.2.9.4 STFT spectrum synthesis 71
8.2.9.5 Inverse FFT 71
8.2.10 Overlap-Add 72
History 73

Intellectual Property Rights

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (http://webapp.etsi.org/IPR/home.asp).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Foreword

This ETSI Standard (ES) has been produced by ETSI Technical Committee Speech Processing, Transmission and Quality Aspects (STQ), and is now submitted for the ETSI standards Membership Approval Procedure.

Introduction

The performance of speech recognition systems receiving speech that has been transmitted over mobile channels can be significantly degraded when compared to using an unmodified signal. The degradations are as a result of both the low bit rate speech coding and channel transmission errors. A Distributed Speech Recognition (DSR) system overcomes these problems by eliminating the speech channel and instead using an error protected data channel to send a parameterized representation of the speech, which is suitable for recognition. The processing is distributed between the terminal and the network. The terminal performs the feature parameter extraction, or the front-end of the speech recognition system. These features are transmitted over a data channel to a remote "back-end" recognizer. The end result is that the transmission channel does not affect the recognition system performance and channel invariability is achieved. ES 201 108 [1] specifies the mel-cepstrum Front-End (FE) to ensure compatibility between the terminal and the remote recognizer.

For some applications, it may be necessary to reconstruct the speech waveform at the back-end. Examples include:

- Interactive Voice Response (IVR) services based on the DSR of "sensitive" information, such as banking and brokerage transactions. DSR features may be stored for future human verification purposes or to satisfy legal requirements.
- Human verification of utterances in a speech database collected from a deployed DSR system. This database can then be used to retrain and tune models in order to improve system performance.
- Applications where machine and human recognition are mixed (e.g. human assisted dictation).

In order to enable the reconstruction of speech waveform at the back-end, additional parameters such as fundamental frequency (F0) and voicing class need to be extracted at the front-end, compressed, and transmitted. The availability of tonal parameters (F0 and voicing class) is also useful in enhancing the recognition accuracy of tonal languages, e.g. Mandarin, Cantonese, and Thai.

The present document specifies a proposed standard for an Extended Front-End (XFE) that extends the Mel-Cepstrum front-end with additional parameters, viz., fundamental frequency F0 and voicing class. It also specifies the back-end speech reconstruction algorithm using the transmitted parameters.

1 Scope

The present document specifies algorithms for extended front-end feature extraction, their transmission, back-end pitch tracking and smoothing, and back-end speech reconstruction which form part of a system for distributed speech recognition. The specification covers the following components:

a) the algorithm for front-end feature extraction to create Mel-Cepstrum parameters;

b) the algorithm for extraction of additional parameters, viz., fundamental frequency F0 and voicing class;

c) the algorithm to compress these features to provide a lower data transmission rate;

d) the formatting of these features with error protection into a bitstream for transmission;

e) the decoding of the bitstream to generate the front-end features at a receiver together with the associated algorithms for channel error mitigation;

f) the algorithm for pitch tracking and smoothing at the back-end to minimize pitch errors;

g) the algorithm for speech reconstruction at the back-end to synthesize intelligible speech.

NOTE: The components (a), (c), (d), and (e) are already covered by the ES 201 108 [1]. Besides these (four) components, the present document covers the components (b), (f), and (g) to provide back-end speech reconstruction and enhanced tonal language recognition capabilities. If these capabilities are not of interest, the reader is better served by (un-extended) ES 201 108 [1].

The present document does not cover the "back-end" speech recognition algorithms that make use of the received DSR front-end features.

The algorithms are defined in a mathematical form, pseudo-code, or as flow diagrams. Software implementing these algorithms written in the 'C' programming language will be provided with the final published version of the present document. Conformance tests are not specified as part of the standard. The recognition performance of proprietary implementations of the standard can be compared with those obtained using the reference 'C' code on appropriate speech databases.

It is anticipated that the DSR bitstream will be used as a payload in other higher level protocols when deployed in specific systems supporting DSR applications.

The Extended Front-End (XFE) standard incorporates tonal information, viz., fundamental frequency F0 and voicing class, as additional parameters. This information can be used for enhancing the recognition accuracy of tonal languages, e.g. Mandarin, Cantonese, and Thai.

The Extended Front-End (XFE) standard incorporates Voice Activity information as part of the voicing class information. This can be used for segmentation (or end-point detection) of the speech data for improved recognition performance.

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 and/or edition number or version number) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies.

Referenced documents which are not found to be publicly available in the expected location might be found at http://docbox.etsi.org/Reference.

[1] ETSI ES 201 108: "Speech Processing, Transmission and Quality Aspects (STQ); Distributed speech recognition; Front-end feature extraction algorithm; Compression algorithms".

[2] ETSI EN 300 903: "Digital cellular telecommunications system (Phase 2+); Transmission planning aspects of the speech service in the GSM Public Land Mobile Network (PLMN) system (GSM 03.50)".

3 Definitions, symbols and abbreviations

3.1 Definitions

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

analog-to-digital conversion: electronic process in which a continuously variable (analog) signal is changed, without altering its essential content, into a multi-level (digital) signal

DC offset: Direct Current (DC) component of the waveform signal

discrete cosine transform: process of transforming the log filterbank amplitudes into cepstral coefficients

fast Fourier transform: fast algorithm for performing the discrete Fourier transform to compute the spectrum representation of a time-domain signal

feature compression: process of reducing the amount of data to represent the speech features calculated in feature extraction

feature extraction: process of calculating a compact parametric representation of speech signal features which are relevant for speech recognition

NOTE: The feature extraction process is carried out by the front-end algorithm.

feature vector: set of feature parameters (coefficients) calculated by the front-end algorithm over a segment of speech waveform

framing: process of splitting the continuous stream of signal samples into segments of constant length to facilitate blockwise processing of the signal

frame pair packet: combined data from two quantized feature vectors together with 4 bits of CRC

front-end: part of a speech recognition system which performs the process of feature extraction

magnitude spectrum: absolute-valued Fourier transform representation of the input signal

multiframe: grouping of multiple frame vectors into a larger data structure

mel-frequency warping: process of non-linearly modifying the scale of the Fourier transform representation of the spectrum

mel-frequency cepstral coefficients: cepstral coefficients calculated from the mel-frequency warped Fourier transform representation of the log magnitude spectrum

notch filtering: filtering process in which the otherwise flat frequency response of the filter has a sharp notch at a pre-defined frequency

NOTE: In the present document, the notch is placed at the zero frequency, to remove the DC component of the signal.

offset compensation: process of removing DC offset from a signal

pre-emphasis: filtering process in which the frequency response of the filter has emphasis at a given frequency range

NOTE: In the present document, the high-frequency range of the signal spectrum is pre-emphasized.

sampling rate: number of samples of an analog signal that are taken per second to represent it digitally

windowing: process of multiplying a waveform signal segment by a time window of given shape, to emphasize pre-defined characteristics of the signal

zero padding: method of appending zero-valued samples to the end of a segment of speech samples for performing a FFT operation

3.2 Symbols

For the purposes of the present document, the following symbols apply:

For feature extraction (clause 4):

bin_k	absolute value of complex-valued FFT output vector *k*
C_i	*i*th cepstral coefficient
$cbin_i$	Center frequency of the *i*th Mel channel in terms of FFT bin indices
$fbank_k$	output of Mel filter for filter bank *k*
f_{c_i}	Center frequency of the *i*th Mel channel
f_i	log filter bank output for the *i*th Mel channel
f_s	input signal sampling rate
f_{s1}, f_{s2}, f_{s3}	symbols for specific input signal sampling rates (8 kHz, 11 kHz, 16 kHz)
f_{start}	starting frequency of Mel filter bank
FFTL	Length of FFT block
$\ln(\ )$	natural logarithm operation
$\log_{10}(\ )$	10-base logarithm operation
M	frame shift interval
$Mel\{\ \}$	Mel scaling operator
$Mel^{-1}\{\ \}$	inverse Mel scaling operator
N	frame length
$round\{\ \}$	operator for rounding towards nearest integer
s_{in}	input speech signal
s_{of}	offset-free input speech signal
s_{pe}	speech signal after pre-emphasis operation
s_w	windowed speech signal

For compression (clause 5):

$idx^{i,i+1}(m)$	codebook index
m	framenumber
$N^{i,i+1}$	compression: size of the codebook

$Q^{i,i+1}$	compression codebook
$q_j^{i,i+1}$	jth codevector in the codebook $Q^{i,i+1}$
$W^{i,i+1}$	weight matrix
y(m)	Feature vector with 14 components

For error mitigation:

$badframeindex_i$	indicator if received VQ index is suspected to be received with transmission error
T_i	threshold on cepstral coefficient

3.3 Abbreviations

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

ADC	Analog-to-Digital Conversion
APM	All-Pole spectral envelope Modelling
CLS	CLaSsification
COMB	COMBined magnitudes estimate calculation
CRC	Cyclic Redundancy Code
CTM	Cepstra To Magnitudes transformation
DC	Direct Current
DCT	Discrete Cosine Transform
DSR	Distributed Speech Recognition
EC	Energy Computation
FE	Front-End
FFT	Fast Fourier Transform (only magnitude components)
HOC	High Order Cepstra
HOCR	High Order Cepstra Recovery
HSI	Harmonic Structure Initialization
IVR	Interactive Voice Response
LBND	Low-Band Noise Detection
LOG	nonlinear transformation
logE	log-Energy measure computation
LSB	Least Significant Bit
LSTD	Line Spectrum to Time-Domain transformation
MF	Mel-Filtering
MFCC	Mer-Frequency Cepstral Coefficient
MSB	Most Significant Bit
Offcom	Offset compensation
OLA	OverLap-Add
PE	Pre-Emphasis
PF	PostFiltering
PITCH	PITCH estimation
PP	Pre-Processing for pitch and class estimation
PTS	Pitch Tracking and Smoothing
SFEQ	Solving Front-EQuation
SNR	Signal-to-Noise Ratio
SS	Spectral Scores
STFT	Short Time Fourier Transform
UPH	Unvoiced PHase
VAD	Voice Activity Detection
VPH	Voiced PHase synthesis
VQ	Vector Quantizer
W	Windowing
XFE	eXtended Front-End

4 Front-end feature extraction algorithm

4.1 Introduction

This clause describes the distributed speech recognition front-end algorithm used to extract mel-cepstral coefficients, fundamental frequency, and voicing class. The specification covers the computation of feature vectors from speech waveforms sampled at different rates (8 kHz, 11 kHz and 16 kHz).

The feature vectors consist of 13 static cepstral coefficients, a log-energy coefficient, a pitch period, and a voicing class.

The feature extraction algorithm defined in this clause forms a generic part of the specification while clauses 4 to 6 define the feature compression and bit-stream formatting algorithms which may be used in specific applications.

The characteristics of the input audio parts of a DSR terminal will have an effect on the resulting recognition performance at the remote server. Developers of DSR speech recognition servers can assume that the DSR terminals will operate within the ranges of characteristics as specified in EN 300 903 [2]. DSR terminal developers should be aware that reduced recognition performance may be obtained if they operate outside the recommended tolerances.

4.2 Front-end algorithm description

4.2.1 Front-end block diagram

Figure 4.1 shows the different blocks of the front-end algorithm.

The details of the Analog-to-Digital Conversion (ADC) are not subject to the present document, but the block has been included to account for the different sampling rates. The blocks Feature Compression and Bit Stream Formatting are covered in clauses 4 to 6 of the present document.

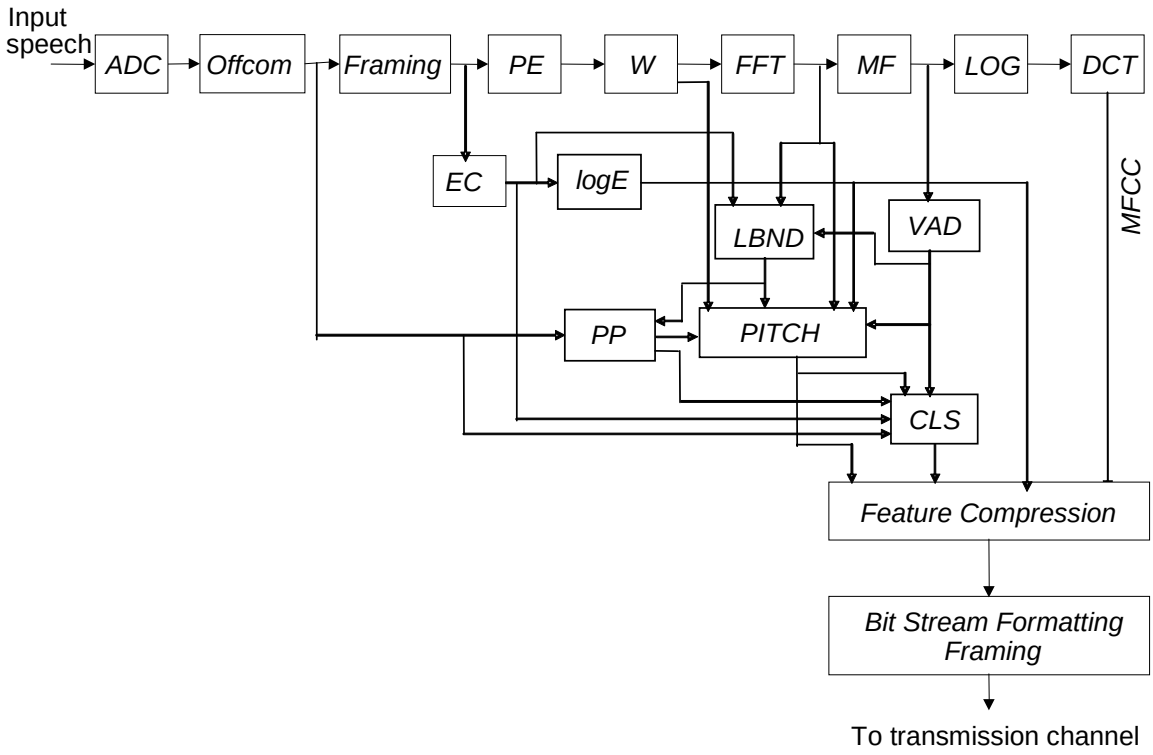


Figure 4.1: Block diagram of the front-end algorithm

4.2.2 Analog-to-digital conversion

The specifics of the analog-to-digital conversion are not part of the present document. Different word-lengths can be used depending on the application.

The output sampling rates of the ADC block are f_{s1} = 8 kHz, f_{s2} = 11 kHz, and f_{s3} = 16 kHz.

4.2.3 Offset compensation

Prior to the framing, a notch filtering operation is applied to the digital samples of the input speech signal s_{in} to remove their DC offset, producing the offset-free input signal s_{of} .

$$s_{of}(n) = s_{in}(n) - s_{in}(n-1) + 0{,}999 \times s_{of}(n-1) \tag{4.1}$$

The offset-free input signal s_{of} is fed into the framing block, the pre-processing block for pitch and class estimation, and the classification block.

4.2.4 Framing

The offset-free input signal s_{of} is divided into overlapping frames of N samples. The frame shift interval (difference between the starting points of consecutive frames) is M samples. The parameter M defines the number of frames per unit time.

The specific values of *N* and *M* depend on the sampling rate according to table 4.1. The frame length is 25 ms for 8 and 16 kHz sampling rates, and 23,27 ms for 11 kHz.

Table 4.1: Values of frame length and frame shift interval depending on the sampling rate

Sampling rate (kHz)	f_{s3} = 16	f_{s2} = 11	f_{s1} = 8
Frame length *N* (samples)	400	256	200
Shift interval *M* (samples)	160	110	80

4.2.5 Energy measure

The frame energy (*E*) and the logarithmic frame energy measure (log*E*) are computed after the offset compensation filtering and framing for each frame:

$$E = \sum_{i=1}^{N} s_{of}(i)^2 \tag{4.2a}$$

$$\log E = \ln(E) \tag{4.2b}$$

Here *N* is the frame length and s_{of} is the offset-free input signal.

A floor is used in the energy calculation, which makes sure that the result for log*E* is not less than -50. The floor value for *E* (lower limit for the argument of ln) is approximately 2e-22.

The frame energy *E* is fed into the low-band noise detection block and the classification block. The log*E* value is fed into the pitch estimation block.

4.2.6 Pre-Emphasis (PE)

A pre-emphasis filter is applied to the framed offset-free input signal:

$$s_{pe}(n) = s_{of}(n) - 0{,}97 \times s_{of}(n-1) \tag{4.3}$$

Here s_{of} and s_{pe} are the input and output of the pre-emphasis block, respectively.

4.2.7 Windowing (W)

A Hamming window of length *N* is applied to the output of the pre-emphasis block:

$$s_w(n) = \left\{0{,}54 - 0{,}46 \times \cos\left(\frac{2\pi(n-1)}{N-1}\right)\right\} \times s_{pe}(n),\ 1 \le n \le N \tag{4.4}$$

Here *N* is the frame length and s_{pe} and s_w are the input and output of the windowing block, respectively. The first sample $s_w(1)$ of each windowed frame, which represents the average spectral value of the corresponding frame, is fed into the pitch estimation block.

4.2.8 Fast Fourier Transform (FFT)

Each frame of *N* samples is zero padded to form an extended frame of 256 samples for 8 kHz and 11 kHz sampling rate, and 512 samples for 16 kHz. An FFT of length 256 or 512, respectively, is applied to compute the short-time Fourier Transform as well as the power and magnitude spectra of the signal:

$$stft_k = \sum s_w(n) e^{-jnk\frac{2\pi}{FFTL}}, \quad k = 0,\ldots,FFTL-1; \tag{4.5a}$$

$$pbin_k = \mathrm{Re}(stft_k)^2 + \mathrm{Im}(stft_k)^2; \tag{4.5b}$$

$$bin_k = \sqrt{pbin_k}, \quad k = 0,\ldots,FFTL-1 \tag{4.5c}$$

Here $s_w(n)$ is the input to the FFT block, *FFTL* is the block length (256 or 512 samples), $stft_k$ is the complex short-time Fourier Transform, $pbin_k$ is the power spectrum, and bin_k is the absolute value of $stft_k$ representing the magnitude spectrum. Both $stft_k$ and bin_k are fed into the pitch estimation block. The magnitude spectrum bin_k is fed into the low-band noise detection block and the mel-filtering block.

NOTE: Due to symmetry, only $stft_{0\ldots FFTL/2}$, $pbin_{0\ldots FFTL/2}$, and $bin_{0\ldots FFTL/2}$ are used for further processing.

4.2.9 Mel-Filtering (MF)

The low-frequency components of the magnitude spectrum are ignored. The useful frequency band lies between 64 Hz and half of the actual sampling frequency. This band is divided into 23 channels equidistant in mel frequency domain. Each channel has triangular-shaped frequency window. Consecutive channels are half-overlapping.

The choice of the starting frequency of the filter bank, f_{start} = 64 Hz, roughly corresponds to the case where the full frequency band is divided into 24 channels and the first channel is discarded using any of the three possible sampling frequencies.

The centre frequencies of the channels in terms of FFT bin indices ($cbin_i$ for the *i*th channel) are calculated as follows:

$$Mel\{x\} = 2\,595 \times \log_{10}\left(1 + \frac{x}{700}\right), \tag{4.6a}$$

$$f_{c_i} = Mel^{-1}\left\{Mel\{f_{start}\} + \frac{Mel\{f_s/2\} - Mel\{f_{start}\}}{23+1} i\right\}, \quad i = 1,\ldots,23, \tag{4.6b}$$

$$cbin_i = round\left\{\frac{f_{c_i}}{f_s} FFTL\right\}, \tag{4.6c}$$

where $round\{\cdot\}$ stands for rounding towards the nearest integer.

The output of the mel filter is the weighted sum of the FFT magnitude spectrum values (bin_i) in each band. Triangular, half-overlapped windowing is used as follows:

$$fbank_k = \sum_{i=cbin_{k-1}}^{cbin_k} \frac{i - cbin_{k-1} + 1}{cbin_k - cbin_{k-1} + 1} bin_i + \sum_{i=cbin_k+1}^{cbin_{k+1}} \left(1 - \frac{i - cbin_k}{cbin_{k+1} - cbin_k + 1}\right) bin_i, \tag{4.7}$$

where k = 1, ..., 23, $cbin_0$ and $cbin_{24}$ denote the FFT bin indices corresponding to the starting frequency and half of the sampling frequency, respectively:

$$cbin_0 = round\left\{\frac{f_{start}}{f_s} FFTL\right\}, \tag{4.8a}$$

$$cbin_{24} = round\left\{\frac{f_s/2}{f_s} FFTL\right\} = FFTL/2. \tag{4.8b}$$

The output of the mel filter is fed into the voice activity detector.

4.2.10 Non-linear transformation

The output of mel filtering is subjected to a logarithm function (natural logarithm).

$$f_i = \ln(fbank_i),\ i = 1, \ldots, 23 \tag{4.9}$$

The same flooring is applied as in the case of energy calculation, that is, the log filter bank outputs cannot be smaller than -50.

4.2.11 Cepstral coefficients

13 cepstral coefficients are calculated from the output of the Non-linear Transformation block.

$$C_i = \sum_{j=1}^{23} f_j \times \cos\left(\frac{\pi \times i}{23}(j - 0{,}5)\right), 0 \le i \le 12 \tag{4.10}$$

4.2.12 Voice Activity Detection (VAD)

The input to the Voice Activity Detection (VAD) block is the mel-filter output $fbank_i$, i = 1, …, 23. The outputs of the *VAD* block are the *vad_flag* and *hangover_flag*. The *vad_flag*, if TRUE, indicates that the current frame is a speech frame. The *hangover_flag*, if TRUE, indicates that the current frame is likely to be a speech frame because it follows a speech segment. The operation of the VAD block is described below with reference to figure 4.4.

In the following, we denote the mel-filter output for the m^{th} frame and i^{th} channel by $F(m,i)$, and when the specific channel is not important, the mel-filter output for the m^{th} frame by $F(m)$. Using these values as input, the channel energy estimator provides a smoothed estimate of the channel energies as follows:

$$E_{ch}(m,i) = \max\left\{E_{\min}, \alpha_{ch}(m) E_{ch}(m-1,i) + (1 - \alpha_{ch}(m))(\lambda_i F(m,i))^2\right\};\ i = 1, 2, \ldots, 23 \tag{4.11}$$

where $E_{ch}(m,i)$ is the smoothed channel energy estimate for the m^{th} frame and the i^{th} channel, E_{min} is the minimum allowable channel energy, $\{\lambda_i, i = 1, 2, \ldots, 23\}$ are the correction factors to compensate for the effect of the pre-emphasis filter and the varying widths of the triangular weighting windows used in mel-filtering, and $\alpha_{ch}(m)$ is the channel energy smoothing factor defined as:

$$\alpha_{ch}(m) = \begin{cases} 0{,}00; & m = 1 \\ 0{,}45; & m > 1 \end{cases} \tag{4.12}$$

The minimum channel energy E_{min} is 5 000 for 8 kHz, 6 400 for 11 kHz, and 10 000 for 16 kHz sampling frequency respectively. he value of the correction factor λ_i is given by the i^{th} value in the 23-element table:

- {3,2811, 2,2510, 1,4051, 1,1038, 0,8867, 0,6487, 0,5482, 0,4163, 0,3234, 0,2820, 0,2505, 0,2036, 0,1680, 0,1397, 0,1179, 0,1080, 0,0931, 0,0763, 0,0674, 0,0636, 0,0546, 0,0478, 0,0046} for 8 kHz;
- {4,1984, 2,6742, 1,9414, 1,5208, 1,0401, 0,8654, 0,7265, 0,4791, 0,4103, 0,3549, 0,2820, 0,2256, 0,1837, 0,1509, 0,1260, 0,1144, 0,0978, 0,0795, 0,0697, 0,0585, 0,0503, 0,0460, 0,0411} for 11 kHz; and
- {3,6723, 2,4663, 1,8348, 1,2653, 0,8148, 0,6619, 0,4988, 0,3912, 0,3080, 0,2325, 0,1891, 0,1476, 0,1171, 0,0990, 0,0809, 0,0621, 0,0522, 0,0444, 0,0362, 0,0301, 0,0256, 0,0222, 0,0194} for 16 kHz.

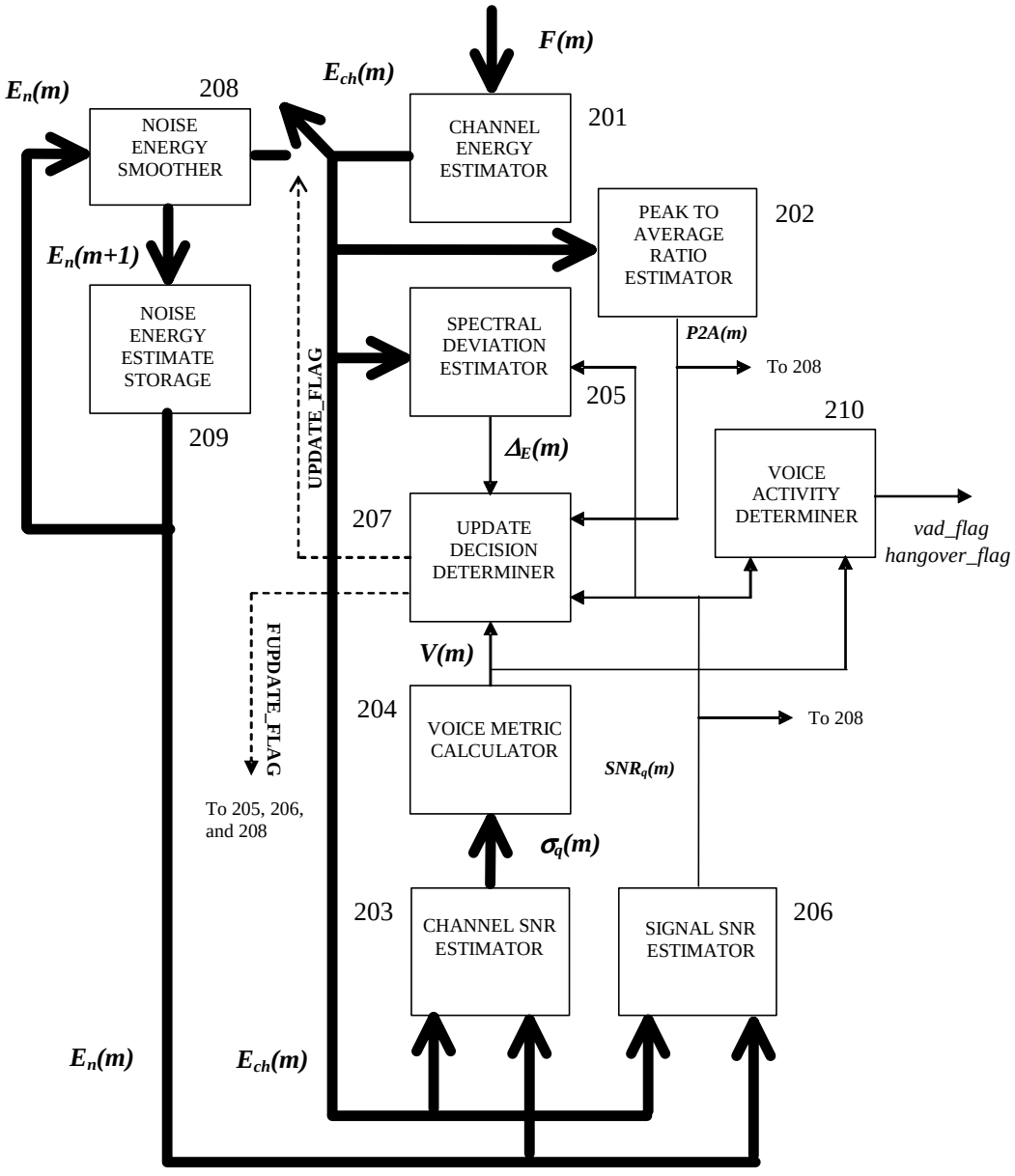


Figure 4.2: Block diagram of the Voice Activity Detection (VAD) algorithm

From the channel energy estimate, the peak-to-average ratio for the current frame *m*, denoted by *P2A*(*m*) is estimated at the peak-to-average ratio estimator as follows:

$$P2A(m) = 10\log_{10}\left(\frac{\max(E_{ch}(m,i)|_{i=5}^{23})}{(1/23)\sum_{i=1}^{23}E_{ch}(m,i)}\right) \tag{4.13}$$

Similar to the channel energy estimate, the channel noise energy estimate (defined below) is initialized as follows:

```
if ((m ≤ INIT_FRAMES) OR (fupdate_flag == TRUE))
{
    if (P2A(m) ≤ PEAK_TO_AVE_THLD)
    {
```

$$E_n(m,i) = \begin{cases} E_{ch}(m,i); & m = 1,\ 1 \le i \le 23; \\ 0{,}7E_n(m-1,i) + 0{,}3E_{ch}(m,i); & 2 \le m \le INIT_FRAMES,\ 1 \le i \le 23; \end{cases}$$

```
    }
    else
    {
```

$$E_n(m,i) = E_{\min};1 \le i \le 23;$$

```
    }
  }
```

(4.14)

where $E_n(m,i)$ is the smoothed noise energy estimate for the m^{th} frame and the i^{th} channel, *INIT_FRAMES* is the number of initial frames which are assumed to be noise-only frames, and *fupdate_flag* is the forced update flag defined later. The value of *INIT_FRAMES* = 10, and that of *PEAK_TO_AVE_THLD* = 10,0. Initially, *fupdate_flag* is set to FALSE.

The channel energy estimate $E_{ch}(m)$ and the channel noise energy estimate $E_n(m)$ are used to estimate the quantized channel Signal-to-Noise Ratio (SNR) indices at the channel SNR estimator as:

$$\sigma_q(m,i) = \max(0, \min(89, \text{round}(10\log_{10}\left(\frac{E_{ch}(m,i)}{E_n(m,i)}\right) \Big/ 0{,}375)));1 \le i \le 23 \tag{4.15}$$

where the values $\{\sigma_q(m, i), i = 1, 2, \ldots, 23\}$ are constrained to be between 0 and 89 both inclusive.

From the channel SNR estimate $\sigma_q(m)$ for the current frame, the voice metric $V(m)$ for the current frame is computed at the voice metric calculator as the sum:

$$V(m) = \sum_{i=1}^{23} v(\sigma_q(i)) \tag{4.16}$$

where $v(k)$ is the k^{th} value of the 90-element voice metric table v defined as: v = {1,1,1,1,1,1,2,2,2,2,2,3,3,3,3,3,4,4,4,4,5,5,5,6,6,7,7,7,8,8,9,9,10,10,11,12,12,13,13,14,15,15,16,17,17,18,19,20,20,21,22,23,24,24,25,26,27,28,28,29,30,31,32,33,34,35,36,37,37,38,39,40,41,42,43,44,45,46,47,48,49,50,50,50,50,50,50,50,50,50,50}.

The channel energy estimate $E_{ch}(m)$ is also used as input to the spectral deviation estimator, which estimates the spectral deviation $\Delta_E(m)$ for the current frame as follows. First, the log energy spectrum is estimated as:

$$E_{dB}(m,i) = 10\log_{10}(E_{ch}(m,i)) ;\ i = 1, 2, \ldots, 23 \tag{4.17}$$

Next, the spectral deviation $\Delta_E(m)$ is estimated as the sum of the absolute difference between the current log energy spectrum and an average long-term log energy spectrum denoted by $\overline{E}_{dB}(m)$, that is:

$$\Delta_E(m) = \sum_{i=1}^{23} \left| E_{dB}(m,i) - \overline{E}_{dB}(m,i) \right| \tag{4.18}$$

The average long-term log energy spectrum is initialized as follows:

if ((*m* ≤ *INIT_FRAMES*) OR (*fupdate_flag* == TRUE))

$$\overline{E}_{dB}(m,i) = E_{dB}(m,i);\ 1 \le i \le 23; \tag{4.19}$$

The average long-term log energy spectrum is updated as follows:

$$\overline{E}_{dB}(m+1,i) = \begin{cases} 0{,}9\overline{E}_{dB}(m,i) + 0{,}1E_{dB}(m,i);\ V(m) > SIG_THLD(m) \\ 0{,}7\overline{E}_{dB}(m,i) + 0{,}3E_{dB}(m,i);\ V(m) \le SIG_THLD(m) \end{cases} \tag{4.20}$$

where the parameter *SIG_THLD*(*m*) depends on the quantized signal SNR described next. The initial value of *SIG_THLD* is 217.

The speech signal SNR is estimated at the signal SNR estimator as follows. First, the total noise energy of the current frame $E_{tn}(m)$ is computed as the sum of the channel noise energies, that is:

$$E_{tn}(m) = \sum_{i=1}^{23} E_n(m,i) \tag{4.21}$$

Next, the instantaneous total signal energy $E_{ts,inst}(m)$ is computed as follows:

if (*V*(*m*) > *SIG_THLD*(*m*))

$$E_{ts,inst}(m) = \sum_{i=1}^{23} \max(E_{ch}(m,i), E_n(m,i));$$

else

$$E_{ts,inst}(m) = E_{tn}(m);$$

end (4.22)

Initialization of $E_{ts,inst}(m)$ is performed as follows:

```
if ((m ≤ INIT_FRAMES) OR (fupdate_flag == TRUE))
```

$$E_{ts,inst}(m) = INIT_SIG_ENRG; \tag{4.23}$$

where the value of *INIT_SIG_ENRG* = 1,0E+09 for 8 kHz, 1,67E+09 for 11 kHz, and 3,0E+09 for 16 kHz respectively.

Once the total instantaneous signal energy and the total noise energy are computed, the instantaneous signal-to-noise ratio of the current frame denoted by $SNR_{inst}(m)$ is computed as:

$$SNR_{inst} = \max(0,0,\ 10\log_{10}(E_{ts,inst}(m) / E_{tn}(m))) \tag{4.24}$$

From the instantaneous SNR, the smoothed SNR is estimated as:

```
if ((m ≤ INIT_FRAMES) OR (fupdate_flag == TRUE))
    SNR(m) = SNR_inst(m);
else
{
    if (V(m) > SIG_THLD(m))
    {
        SNR(m) = β SNR(m-1) + (1-β) SNR_inst(m);
        β = min(β+0.003, HI_BETA);
    }
    else
        β = max(β-0.003, LO_BETA);
}
```
(4.25)

The lower and upper limits of the smoothing factor β are respectively *LO_BETA* = 0,950 and *HI_BETA* = 0,998. Initially, the value of β is set at *LO_BETA*. The signal SNR is then quantized to 20 different values as:

$$SNR_q(m) = \max(0,\min(\text{round}(SNR(m)/1{,}5),19)) \tag{4.26}$$

The quantized signal SNR is used to determine different threshold values. For example, the signal threshold for the next frame *SIG_THLD*(*m*+1) is determined using $SNR_q(m)$ as an index into the 20-element table {36, 43, 52, 62, 73, 86, 101, 117, 134, 153, 173, 194, 217, 242, 268, 295, 295, 295, 295, 295}.

At this point, the voice metric $V(m)$, the spectral deviation $\Delta_E(m)$, the peak-to-average ratio $P2A(m)$, and the quantized signal SNR $SNR_q(m)$ are input to an update decision determiner. The logic shown below in pseudo-code demonstrates how the noise estimate update decision is made and also how a forced update decision is made (a forced update mechanism allows the voice activity detector to recover from wrong classification of background noise as speech whenever there is a sudden increase in background noise level).

First, the update threshold for the current *frame UPDATE_THLD*(*m*) is determined using $SNR_q(m)$ as an index into a 20-element table given by {31, 32, 33, 34, 35, 36, 37, 37, 37, 37, 37, 37, 37, 37, 37, 38, 38, 38, 38, 38}. The update decision determination process begins by clearing the update flag (*update_flag*) and the forced update flag (*fupdate_flag*). These flags are set if certain conditions are satisfied as illustrated by the pseudo-code below. The initial value of *update_cnt* is set to 0.

```
update_flag = FALSE;
fupdate_flag = FALSE;
if ((m > INIT_FRAMES) AND (V(m) < UPDATE_THLD(m)) AND
    (P2A(m) < PEAK_TO_AVE_THLD)
{
    update_flag = TRUE;
    update_cnt = 0;
}
else
{
    if ((P2A(m) < PEAK_TO_AVE_THLD) AND (Δ_E(m) < DEV_THLD))
    {
        update_cnt = update_cnt + 1;
        if (update_cnt ≥ UPDATE_CNT_THLD)
        {
            update_flag = TRUE;
            fupdate_flag = TRUE;
        }
    }
}
```

(4.27)

In order to avoid long term "creeping" of the update counter (*update_cnt*) setting the forced update flag (fupdate_flag) falsely in the above pseudo-code, an hysteresis logic is implemented as shown below. Initial values of *last_update_cnt* and *hyster_cnt* are set to 0.

```
if (update_cnt == last_update_cnt)
        hyster_cnt = hyster_cnt + 1;
    else
    {
        hyster_cnt = 0;
        last_update_cnt = update_cnt;
    }
    if (hyster_cnt > HYSTER_CNT_THLD)
        update_cnt = 0;
```

(4.28)

The values of different constants used above are as follows: *DEV_THLD* = 70, *UPDATE_CNT_THLD* = 500, and *HYSTER_CNT_THLD* = 9. Whenever the above referenced update flag is set for a given frame, the channel noise estimate for the next frame is updated in the noise energy smoother as follows:

$$E_n(m+1,i) = 0{,}9E_n(m,i) + 0{,}1E_{ch}(m,i)) \text{ ; } i = 1, 2, \ldots, 23 \quad (4.29)$$

The updated channel noise estimate is stored in noise energy estimate storage for all future frames until the next update occurs. The output of the noise energy estimate storage $E_n(m)$ is used as an input to the channel SNR estimator as described earlier.

Next, we describe the operation of the voice activity determiner, which uses the voice metric $V(m)$ and the quantized signal SNR value $SNR_q(m)$ as inputs. For the first *INIT_FRAMES* frames, the outputs of the voice activity determiner, viz., *vad_flag* and *hangover_flag* are set to FALSE since these frames are assumed to be noise-only frames. For the following frames, the voice activity determiner operates by testing if the voice metric exceeds the voice metric threshold V_{th} If the output of this test is TRUE, then the current frame is declared "voice-active". Otherwise, the hangover count variable (*hangover_count*) is tested to find out if it is greater than or equal to zero. If the output of this test is TRUE, then also the current frame is declared "voice-active". If the outputs of both tests are FALSE, then the current frame is declared "voice-inactive". The "hangover" mechanism is generally used to cover slowly decaying speech that might otherwise be classified as noise, and to bridge over small gaps or pauses in speech. It is activated if the number of consecutive "voice-active" frames (counted by the *burst_count* variable) is at least equal to B_{cnt}, the burst count threshold. To activate the mechanism, the number of hangover frames is set to H_{cnt}, the hangover count threshold. The pseudo-code for the voice activity determiner is shown below. To begin with, the voice metric threshold V_{th}, the hangover count threshold H_{cnt}, and the burst count threshold B_{cnt} are initialized to 56, 28 and 6 respectively. Furthermore, the variables hangover_count and burst_count are both initialized to 0.

```
if (V(m) > Vth(m))
    {
        vad_local = TRUE;
        burst_count = burst_count + 1;
        if (burst_count >= Bcnt(m))
            hangover_count = Hcnt(m);
    }
    else
    {
        vad_local = FALSE:
        burst_count = 0;
    }

if ((vad_local == TRUE) OR (hangover_count > 0))
        vad_flag = TRUE;
    else
        vad_flag = FALSE;

if ((vad_local == FALSE) && (hangover_count > 0))
{
        hangover_flag = TRUE;
        hangover_count = hangover_count - 1;
    }
    else
        hangover_flag = FALSE;                                    (4.30)
```

As a final step, the quantized SNR value is used to determine the voice metric threshold V_{th}, the hangover count threshold H_{cnt}, and the burst count threshold B_{cnt} for the next frame as:

$$V_{th}(m+1) = V_{table}[SNR_q(m)],\ H_{cnt}(m+1) = H_{table}[SNR_q(m)],\ B_{cnt}(m+1) = B_{table}[SNR_q(m)] \qquad (4.31)$$

where $SNR_q(m)$ is used as an index into the respective tables. These tables are defined by: V_{table} = {32, 34, 36, 38, 40, 42, 44, 46, 48, 50, 52, 54, 55, 56, 57, 57, 58, 58, 58, 58}, H_{table} = {54, 52, 50, 48, 46, 44, 42, 40, 38, 36, 34, 32, 30, 28, 26, 24, 22, 20, 18, 16}, and B_{table} = {2, 2, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6}.

4.2.13 Low-Band Noise Detection (LBND)

The input to the Low-Band Noise Detection (LBND) block are the power spectrum $pbin_k$, $k=0,\ldots,FFTL/2$, the *vad_flag* and the frame energy E. The output of the LBND block is *lbn_flag* indicating (if TRUE) that the current frame contains background noise in the low frequency band.

The *LBND* code maintains an internal state variable *LH_Ratio* which is initialized to 1,9. The operation of the *LBND* block is described by the following pseudo code wherein the *cut_idx* parameter is defined as:

$$cut_idx = floor\left(380 \times FFTL / \left(1000 \times f_s\right)\right) \qquad (4.32)$$

```
if (vad_flag == FALSE)
{
    if (2E/FFTL < 500)
        cur_ratio = 0;
    else
{
    low_max =  max_{1≤k≤cut_idx} pbin_k ;

    high_max =  max_{cut_idx<k≤FFTL/2} pbin_k ;
    if (high_max == 0)
        cur_ratio = 10;
    else
        cur_ratio = low_max / high_max
}
    LH_Ratio = 0,99×LH_Ratio + 0,01×cur_ratio;
}
    if (LH_Ratio > 1,9)
        lbn_flag = TRUE;
    else
        lbn_flag = FALSE;                                                    (4.33)
```

4.2.14 Pre-Processing for pitch and class estimation

The input to the Pre-Processing (PP) block is the offset-free input signal s_{of} from the *Offcom* block and the *lbn_flag* from the Low-Band Noise Detection (LBND) block. The outputs of the *PP* block are the low-pass filtered, downsampled speech signal s_{lpds} which is fed into the Pitch estimation block (PITCH) and the high-pass filtered upper-band signal s_{ub} which is fed into the Classification block (CLS). The low-pass and high-pass filtering are performed using pole-zero filters with the generic form shown below:

$$y(n) = b_0 x(n) + b_1 x(n-1) + ... + b_M x(n-M) - a_1 y(n-1) - a_2 y(n-2) - ... - a_M y(n-M) \quad (4.34)$$

where x is the input, y is the output, M is order of the filter, $b_0, b_1, ..., b_M$ are the coefficients of the numerator polynomial defining the zeros, and $1, a_1, a_2, ..., a_M$ are the coefficients of the denominator polynomial defining the poles. The filter coefficients used are shown in table 4.2. The low-pass filtered speech is first decimated by a factor *DSMP*, where *DSMP* is 4 for 8 kHz, 5 for 11 kHz, and 8 for 16 kHz respectively. The latest (2×*MAX_PITCH* / *DSMP*) samples referred to as the *low-pass filtered extended downsampled frame* is fed into the *PITCH* block. The value of the *MAX_PITCH* parameter is 160 for 8 kHz, 220 for 11 kHz and 320 for 16 kHz respectively.

Table 4.2: Filter coefficients used in the pre-processing block

Samp. frequency / Filter details	8 kHz	11 kHz	16 kHz
low-pass filter numerator coefficients filter order - 7 *lbn_flag* = FALSE	0,0003405377 0,0018389033 0,0038821292 0,0037459142 0,0010216130 -0,0010216130 -0,0008853979 -0,0002043226	0,00006475945579 0,00034263580465 0,00069586625626 0,00060637516431 0,00005297323484 -0,00030025721678 -0,00021076612482 -0,00004592093010	0,00000857655707 0,00004459809678 0,00008748088215 0,00006861245659 -0,00000857655707 -0,00005145934244 -0,00003259091688 -0,00000686124566
low-pass filter denominator coefficients; filter order - 7 *lbn_flag* = FALSE	1,00000000 -4,47943480 8,88015848 -10,05821568 6,99836861 -2,98181953 0,71850318 -0,07538083	1,00000000000000 -5,16457301342956 11,60327150757658 -14,68045002998683 11,28039703154784 -5,25795344738947 1,37514936680065 -0,15553999870817	1,00000000000000 -5,73713549885214 14,19729645263144 -19,63612073482969 16,38673682892475 -8,24809503698812 2,31775924387808 -0,28041380978170

Samp. frequency / Filter details	8 kHz	11 kHz	16 kHz
low-pass filter numerator coefficients filter order - 6 *lbn_flag* = TRUE	0,00034054 0,00204323 0,00510806 0,00681075 0,00510806 0,00204323 0,00034054	0,00006475945579 0,00038855673475 0,00097139183688 0,00129518911584 0,00097139183688 0,00038855673475 0,00006475945579	0,00000857655707 0,00005145934244 0,00012864835610 0,00017153114146 0,00012864835610 0,00005145934244 0,00000857655707
low-pass filter denominator coefficients filter order - 6 *lbn_flag* = TRUE	1,00000000 -3,57943480 5,65866717 -4,96541523 2,52949491 -0,70527411 0,08375648	1,00000000000000 -4,23729801342957 7,67413099217370 -7,56442021421899 4,26609927740795 -1,30210623993103 0,16773880316861	1,00000000000000 -4,78713549885213 9,64951772872192 -10,46907889254388 6,44111188100808 -2,12903875003046 0,29517243134916
high-pass filter numerator coefficients filter order - 6	0,14773250 -0,88639500 2,21598750 -2,95464999 2,21598749 -0,88639500 0,14773250	0,25710908848444 -1,54265453090663 3,85663632726659 -5,14218176968878 3,85663632726659 -1,54265453090663 0,25710908848444	0,39802968073138 -2,38817808438830 5,97044521097075 -7,96059361462766 5,97044521097075 -2,38817808438830 0,39802968073138
high-pass filter denominator coefficients filter order - 6	1,00000000 -2,37972104 2,91040657 -2,05513144 0,87792390 -0,20986545 0,02183157	1,00000000000000 -3,36067979080750 5,06982907485034 -4,27873732337721 2,10853144888207 -0,57109866030671 0,06610536478028	1,00000000000000 -4,18238957916850 7,49161108458765 -7,31359596689075 4,08934993183312 -1,23852537177671 0,15842763255178

4.2.15 Pitch estimation

A flowchart of the pitch estimation process is shown on figure 4.3. Pitch frequency (*F0*) candidates are generated sequentially in high, middle and low frequency intervals (*search ranges*). The candidates generated for a search range are added to the candidates generated earlier and an attempt is made to determine a pitch estimate among the candidates. If the pitch estimate is not determined, the next search range is processed. Otherwise certain internal variables, which represent the pitch estimation history information are updated. At output, the pitch estimate is converted from the frequency to time representation or is set to 0 indicating an unvoiced frame.

4.2.15.1 Dirichlet interpolation

Frequency resolution of the discrete complex spectrum in the diapason [0 kHz, 4 kHz] is doubled by the interpolation of the STFT (4.5a) by Dirichlet kernel. The interpolated STFT is calculated as follows:

$$
\begin{aligned}
& istft(2n) = stft(n) \\
& \mathrm{Re}[istft(2n+1)] = \mathrm{s_w}(1) - \sum_{k=0}^{LDK-1} D(k) \times \{\mathrm{Im}[stft(n-k)] - \mathrm{Im}[stft(n+1+k)]\} \\
& \mathrm{Im}[istft(2n+1)] = \sum_{k=0}^{LDK-1} D(k) \times \{\mathrm{Re}[stft(n-k)] - \mathrm{Re}[stft(n+1+k)]\} \\
& n = 0,1,..., N4kHz
\end{aligned}
\qquad (4.35)
$$

where:

(N4kHz-1) is the index of the FFT point representing 4kHz frequency:

$$D(k) = \frac{1}{FFTL} \Big/ tg\left(\frac{\pi}{FFTL} \times (k + 0{,}5)\right)$$

$$LDK = 8$$

In (4.35), an $stft(i)$ value corresponding to a negative value of $i < 0$ is replaced by the complex conjugate $stft \times (-i)$ associated with $-i$.

The number of *istft* samples computed and used further is $FFTIL = 2 \times N4kHz - 1$. The *istf* vector is used for the processing of the current and the next frames.

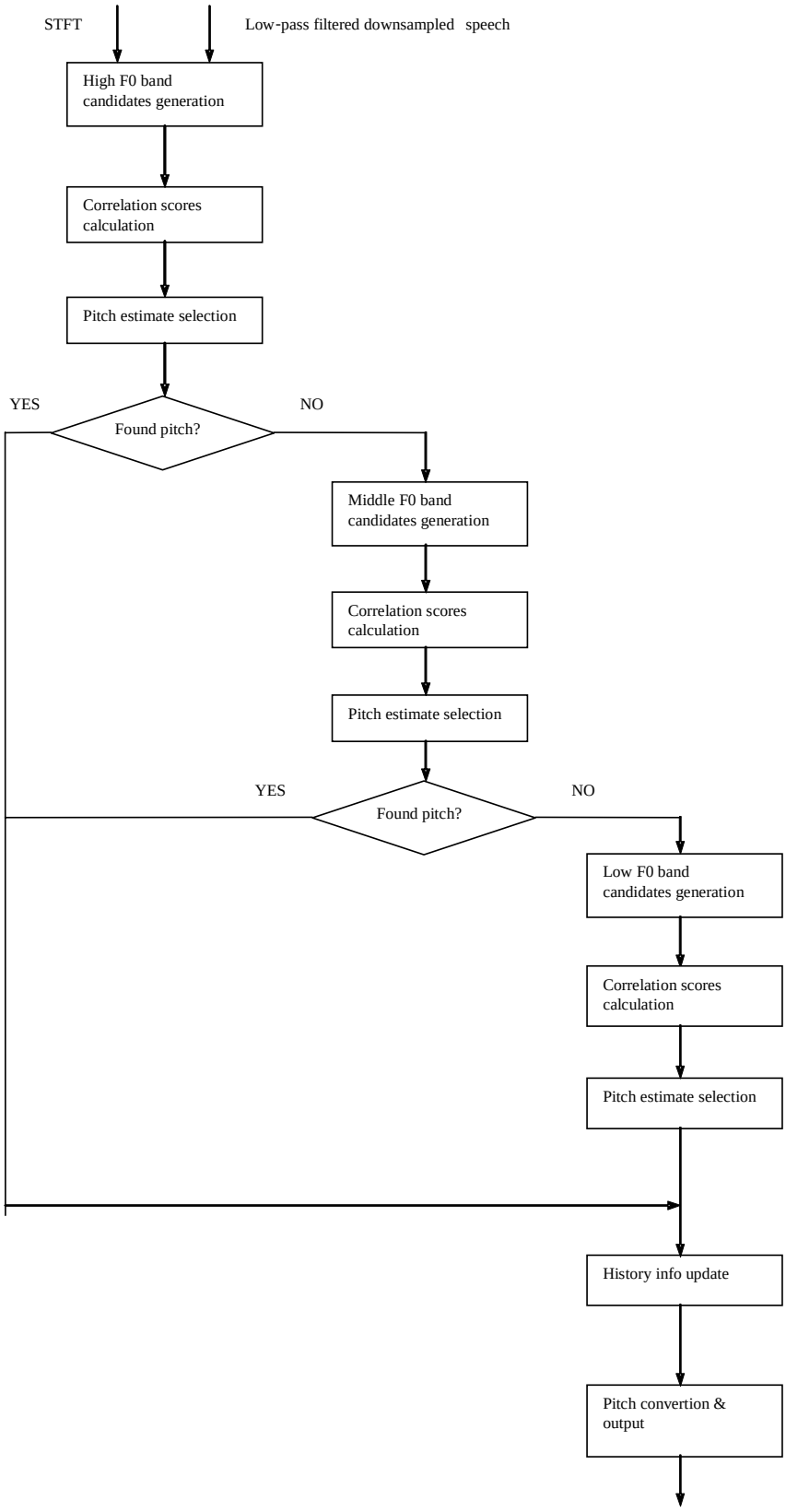


Figure 4.3: Pitch estimation flowchart

4.2.15.2 Non-speech and low-energy frames

If the frame either has been classified by the VAD block as a non-speech frame or its log-energy value is less than a predefined threshold *log E < 13,6* then the pitch frequency F0 estimate is set to 0 and the final step of history information update is performed as described further.

4.2.15.3 Search ranges specification and processing

The entire search diapason for pitch frequency is defined as *SR* = [52 Hz, 420 Hz]. If a variable *StableTrackF0* (which is described below) has a non-zero value then *SR* is narrowed as follows:

SR = SR ∩ [0,666 × StableTrackF0, 2,2 × StableTrackF0].

Three slightly overlapping search ranges are specified:

SR1 = SR ∩ [52 Hz, 120 Hz];

SR2 = SR ∩ [100 Hz, 210 Hz];

SR3 = SR ∩ [200 Hz, 420 Hz]

The processing stages described in clauses 14.2.15.4 to 14.2.15.7 are performed consequently for the three search ranges in the order SR3, SR2, SR1. If there are differences specific to a certain search range they are explained in the relevant clause. It might happen that some of the search ranges are empty. No processing is performed for an empty search range.

4.2.15.4 Spectral peaks determination

This stage is performed only twice: first time for the *SR3* and *SR2* ranges, and a second time for *SR1*.

When the processing is being performed for *SR3*/*SR2* search interval, power spectrum with doubled frequency resolution is computed as follows:

$$ps(n) = \begin{cases} pbin_{n/2}, & for\ even\ \ n \\ \mathrm{Re}[istft(n)]^2 + \mathrm{Im}[istft(n)]^2, & for\ odd\ \ n \end{cases} \tag{4.36}$$

When the processing is being performed for SR1 search interval, an STFT corresponding to a double frame is approximated as follows:

$$istft_2(n) = istft(n) + \exp(-j \times \frac{\pi \times n \times M}{FFTIL}) \times istft_{prev}(n) \tag{4.37}$$

where $istft_{prev}$ is the Dirichlet interpolated STFT of the previous frame. Then power spectrum is computed as:

$$ps(n) = \mathrm{Re}[istft_2(n)]^2 + \mathrm{Im}[istft_2(n)]^2 \tag{4.38}$$

In (4.36) to (4.38), n = 0, 1, …, FFTIL - 1 corresponding to the frequency interval [0 kHz, 4 kHz].

Power spectrum is multiplied by the inverse squared frequency response of the pre-emphasis operator (4.3):

$$dps(n) = ps(n) \times \frac{1}{1 - 2 \times 0{,}97 \cos(\pi \times n / LFFT) + 0{,}97^2} \tag{4.39}$$

and smoothed:

$$\begin{aligned} &sps(n) = 0{,}625 \times dps(n) + 0{,}1875 \times [dps(n-1) + dps(n+1)], \\ &for\ \ n = 1, ..., FFTIL - 2 \\ &sps(0) = dps(0), \quad sps(FFTIL-1) = dps(FFTIL-1) \end{aligned} \tag{4.40}$$

The values of the smoothed power spectrum *sps(n)* are analysed within the range n∈ [N_0+2, FFTIL-3] and all local maxima are determined. N_0 is set to $300 \times 2FFTL/(1\ 000 \times f_s)$ if low band noise has been detected at that frame. Otherwise $N_0 = 0$. That is, if low band noise is present then the spectral components residing at frequencies lower than 300 Hz are not analysed. A value *sps(n)* is considered as a local maximum if the following condition is TRUE:

$$sps(n) > sps(n-1) \wedge sps(n) > sps(n+1) \wedge [sps(n-1) \geq sps(n-2) \vee sps(n+1) \geq sps(n+2)$$

Let *{(A_k, n_k), k = 1, …, Npeaks}* be a list of all the local maxima (representing spectral peaks) sorted in ascending order of their frequencies where $A_k = sps(n_k)$.

Scaling down of high frequency peaks

The entire range [0, FFTIL] of the frequency index is divided into three equal sub-intervals, and the maximal values $Amax_1$, $Amax_2$ and $Amax_3$ of A_k is found in the low, middle and high sub-intervals correspondingly. The value $Amax_j$ *(j = 2,3)* is evaluated against a threshold $THR_j = A\max_1 \times \rho_j^{\,2}$. If $Amax_j > THR_j$ then all the A_k associated with j-th interval are multiplied by factor $THR_j / Amax_j$. The following parameter values are used $\rho_2 = 0{,}65$; $\rho_3 = 0{,}45$.

If the number of the peaks (the local maxima) exceeds 30 then the peaks with amplitudes less than $0{,}001^2 \times \max A_k$ are discarded from the peaks list. If the number of remaining peaks is still exceeds 30 then all the high frequency peaks starting from the peak #31 are discarded. The total number *Npeaks* of the peaks is updated as needed.

The peaks are sorted in descending order of their amplitudes. If the number of peaks is greater than 20 then only 20 first peaks are selected for further processed, and the number *Npeaks* is set to 20.

Location and amplitude of each peak is refined by fitting parabola through the corresponding local maximum and the two neighbouring samples of the power spectrum *sps*.

$$\begin{aligned} &loc_k = n_k - 0{,}5 \times b/a \\ &refA_k = sps(n_k + 1) + 0{,}25 \times b \times (loc_k - n_k), \\ &where \\ &\qquad a = sps(n_k - 1) - 2A_k + sps(n_k + 1),\ and \\ &\qquad b = sps(n_k + 1) - sps(n_k - 1) \end{aligned} \tag{4.41}$$

Then the peak locations loc_k are converted to Hz units and the square roots are taken from the peak amplitudes:

$$\begin{aligned} &PF_k = loc_k \times 1000 \times f_s/(2 \times FFTL) \\ &PA_k = \sqrt{refA_k} \end{aligned} \tag{4.42}$$

The sequence *{PA_k, PF_k, k = 1,…,Npeaks}* represents magnitude spectrum peaks.

Scaling down of high frequency peaks procedure is applied to this peaks sequence as described above except for that this time ρ_j is used for the threshold THR_j computation instead of $\rho_j^{\,2}$.

If *Npeaks* > 7 the final attempt to reduce the number of peaks is done as follows. If a number *N1* exists so that $\sum_{k=1}^{N1} PA_k \leq 0{,}95 \times \sum_{k=1}^{Npeaks} PA_k$ then only *N1* starting peaks are taken. Otherwise the peaks are scanned from the end of the list towards the beginning and all the peaks with amplitudes less than $0{,}406 \times PA_7$ are put out. The number *Npeaks* of peaks is updated.

The peak amplitudes are normalized:

$$NPA_k = PA_k \Big/ \sum_{i=1}^{Npeaks} PA_i \tag{4.43}$$

4.2.15.5 F0 Candidates generation

Pitch candidates are selected among the local maxima of a piecewise constant *utility function U(F0)*:

$$U(F0) = \sum_i NPA_i \times I(PF_i / F0)$$

where

$$I(r) = \begin{cases} 1, & |r| \le D1 \\ 0{,}5, & D1 < |r| \le D2 \\ 0, & D2 < |r| < 0{,}5 \end{cases} \qquad (4.44)$$

$$I(r+1) = I(r)$$

$$D1 = 65/512, \quad D2 = 100/512$$

Lower *F0min* and upper *F0max* limits for F0 are defined as the left and the right edges respectively of the processed search range *SRi, i=1, 2, 3.*

First, a partial utility function is built including only contributions of a few highest peaks. The partial utility function is represented by a list of break points. Then all local maxima locations of the partial utility function are determined. Finally, the values of the whole utility function at the local maxima are computed.

Building partial utility function

NPprelim peaks are selected from the top of the peaks list. *NPprelim = min(Npeaks, 7).* A counter variable is initialized BPCount =0. For each peak *(NPA_k, PF_k), k = 1, …, NPprelim*, a list BPL_k of the utility function break points is collected as described below.

The maximal and minimal dividers of the peak frequency are calculated:

$$N_{\min} = ceil\left[\max(0, \frac{PF_k}{F0\max} - D1) \right] \qquad N_{\max} = floor\left(\frac{PF_k}{F0\min} + D2 \right) \qquad (4.45)$$

The counter BPCount is updated $BPCount = BPCount + N_{max} - N_{min} + 1$ and compared against a predefined threshold BPLimit:

$$BPLimit = \begin{cases} 60 \ for \ SR1 \\ 30 \ for \ SR2 \\ 20 \ for \ SR3 \end{cases} \qquad (4.46)$$

If the counter value exceeds the threshold then the entire peaks processing is terminated, and no more break point lists are built. Otherwise the processing of the k-th peak continues. Index *n* scans the range *[Nmin, Nmax]* in the reverse order *n = Nmax, Nmax - 1, …, Nmin* each time generating four new breakpoints in the list, each break point is given by its frequency value BPF and amplitude value BPA:

$$\begin{aligned} BPF_{4(n-1)+1} &= PF_k/(n + D2) & BPA_{4(n-1)+1} &= 0{,}5 \times PA_k \\ BPF_{4(n-1)+2} &= PF_k/(n + D1) & BPA_{4(n-1)+2} &= 0{,}5 \times PA_k \\ BPF_{4(n-1)+3} &= PF_k/(n - D1) & BPA_{4(n-1)+3} &= -0{,}5 \times PA_k \\ BPF_{4(n-1)+4} &= PF_k/(n - D2) & BPA_{4(n-1)+4} &= -0{,}5 \times PA_k \end{aligned} \qquad (4.47)$$

Note that the break points in the list are ordered in the increasing order of the frequency.

If the list is not empty and $BPF_1 < F0\min$ then the beginning of the list is modified as follows. The first k = max(1, m - 2) elements are discarded where $m = \min i : \{BPF_i > F0\min\}$. The new head of the list (former element #m-1) is set to:

$$BPF = F0\min, \ BPA = \sum_{j=1}^{k+1} BPA_j$$

If the list is not empty and there are elements (at the tail) with $BPF \geq F0\max$, that elements are deleted from the list.

Finally, if $F0\max > PF_k / D2$ then one or two elements are appended at the end of the list depending on certain conditions as described below. Two frequency values are calculated: $F1 = PF_k / D2$ and $F2 = PF_k / D1$.

if ($F2 < F0\min$)

One element is appended: $BPF = F0\min,\ \ BPA = PF_k$

else if ($F1 < F0\min < F2 \leq F0\max$)

Two elements are appended: $BPF = F0\min,\ \ BPA = 0{,}5PF_k$ and $BPF = F2,\ \ BPA = 0{,}5PF_k$

else if ($F1 < F0\min \wedge F2 > F0\max$)

One element is appended: $BPF = F0\min,\ \ BPA = 0{,}5PF_k$

else if ($F1 \geq F0\min \wedge F2 \leq F0\max$)

Two elements are appended: $BPF = F1,\ \ BPA = 0{,}5PF_k$ and $BPF = F2,\ \ BPA = 0{,}5PF_k$

else if ($F1 \geq F0\min \wedge F2 > F0\max$)

One element is appended: $BPF = F1,\ \ BPA = 0{,}5PF_k$

All the break point lists *{BPL_k}* are merged together into one array $U_{partial}$ = *{(BPF_n, BPA_n)}* preserving the frequency ascending order, and the amplitudes of the break points are modified as:

$$BPA_n = BPA_n + BPA_{n-1}, n = 2, 3, \ldots$$

If the last break point frequency is less than *F0max* then a new terminating element *(BPF = F0max, BPA = 0)* is appended to the array. Further we will refer to the number of elements in the $U_{partial}$ array as NBP.

Preliminary candidates determination

NCprelim break points are determined which are the highest in amplitude local maxima among the elements of the $U_{partial}$ array, where *NCprelim = min(4,NBP)*. These break points being sorted in the descended order of amplitude form a *list of preliminary candidates*. If a variable *StableTrackF0* (which is described below in clause 4.2.15.8) has a non-zero value then an additional break point *BPad* is sought which is the highest in amplitude local maximum among the $U_{partial}$ array elements having frequency in the range [*StableTrackF0/1,22, StableTrackF0 × 1,22*]. If such the break point is found then the amplitude associated with it is increased by 0,06 and compared against the amplitudes of the preliminary candidates list members. If the modified amplitude is greater than the amplitude of at least one of the preliminary candidates then *BPad* is inserted into the preliminary candidate list so that the list elements order is preserved, and the last list member is put out. Finally, the frequency value for each candidate is modified as:

$$BF_n = 0{,}5 \times (BF_n + BF_{n+1})$$

If $n < NBP$ where n is the index of the break point in the $U_{partial}$ array.

Candidate amplitudes refinement

For each preliminary candidate the amplitude value is recomputed in accordance to formula (4.44) wherein F0 is substituted by the frequency value associated with that candidate and the summation is performed over all the *Npeaks* spectral peaks.

Final candidates determination

NC (final) candidates are selected from the preliminary candidates, *NC = min(2,Nprelim)*. For the selection purpose a compare function is defined for a pair (*F1*,*A1*) and (*F2*,*A2*) of candidates given by their frequencies *Fi* and amplitudes *Ai*. Let *F1 < F2*. The first candidate is declared to be better than the second one if the following condition is satisfied:

$$A1 > A2 + 0{,}06 \vee (A1 > A2 \wedge 1{,}17 \times F1 > F2) \tag{4.48}$$

otherwise the second candidate is considered as the best between the two.

NC best candidates are determined, sorted in descending order of their quality, and form a final candidates list. If the pitch estimate *PrevF0* obtained at the previous frame has non-zero value then the preliminary candidates are determined having frequency values within the interval [Prev*F0/1,22, PrevF0 × 1,22*]. If such preliminary candidates exist then one of them having the maximal amplitude is declared as an additional candidate. The amplitude *a* of the additional candidate is increased by 0,06 (*b=a+0,06*), and compared against the amplitudes of the final candidates list members. If a member exists with amplitude less than *b* then the last member of the final candidates list is replaced by the additional candidate.

Below the amplitudes associated with the candidates are referred to as Spectral Scores (SS).

4.2.15.6 Computing correlation scores

Correlation score is computed for each pitch candidate. The input for correlation score calculation stage comprises the low-pass filtered extended downsampled frame (clause 4.2.14) and the candidate pitch frequency F0. Here we designate the low-pass filtered extended downsampled frame by *u(n)* and assume that the origin *n = 0* is associated with the sample #NDS counting from the end of the vector *u*, so that the preceding to it samples have negative index values. NDS is the length of downsampled frame (clause 4.2.14) *NDS = N / DSMP* where DSMP is a downsampled factor (clause 4.2.14).

Candidate pitch frequency is converted to a time -domain lag:

$$\tau = \frac{1\,000 \times f_s}{F0 \times DSMP} \tag{4.49}$$

An integer lag is calculated by rounding the lag value to the upper integer number $i\tau = ceil(\tau)$.
Analysis window length is calculated:

$$LW = floor\left(\frac{75 \times f_s}{8 \times DSMP}\right) \tag{4.50}$$

Offset and length parameters calculation

Offset O and length Len parameters are calculated to be used by further processing, besides two following cases are treated differently.

Case 1:

$$i\tau \le LW$$

$$O = i\tau + \arg\max_{0 \le t \le NDS - LW - i\tau} E(t), \text{ where}$$

$$E(t) = \sum_{n=t}^{t+LW+i\tau-1} u(n)^2$$

$$Len = LW + i\tau$$

Case 2:

$$i\tau > LW$$

Two vectors are extracted from the signal *u*:

$$u1=\{u(t0), u(t0 + 1), \ldots, u(t0 + i\tau - 1)\} \text{ and } u2=\{u(t0 - i\tau), u(t0 + 1 - i\tau), \ldots, u(t0 - 1)\},$$

where:

$$t0 = \begin{cases} NDS/2, \text{ if } i\tau < NDS/2 \\ NDS - i\tau, \quad \text{otherwise} \end{cases}$$

An auxiliary offset *ofs* is determined as:

$$ofs = \arg\max_{0 \le t \le i\tau - 1} E(t)$$

where:

$$E(t) = \sum_{n=0}^{LW-1} (u(n0 + t \bmod i\tau + n)^2 + u(n0 + t \bmod i\tau + n - i\tau)^2),$$

$$t0 = \begin{cases} NDS/2, \text{ if } i\tau < NDS/2 \\ NDS - i\tau, \quad \text{otherwise} \end{cases}$$

If:

$$ofs + LW \le i\tau \text{ then } O = t0 + ofs \text{ and } Len = LW.$$

Otherwise two sets of the offset and length parameters are prepared:

$$\{O1 = t0 + ofs, Len1 = i\tau - ofs\} \text{ and } (O2 = t0, Len2 = LW - Len1\}.$$

Correlator

Input parameters for this block are *O*, *Len* and $i\tau$

Three vectors are extracted from *u*:

$X = \{u(O), u(O + 1), \ldots, u(O + Len - 1)\}^T$

$Y = \{u(O - i\tau), u(O - i\tau + 1), \ldots, u(O - i\tau + Len - 1)\}^T$

$Z = \{u(O - i\tau + 1), u(O - i\tau + 2), \ldots, u(O - i\tau + Len)\}^T$

For each vector the sum of the coordinates is computed: ΣX, ΣY and ΣZ. The following inner products are computed also: X^TX, Y^TY, Z^TZ, X^TY, X^TZ and Y^TZ.

Where there are two sets of the offset and length parameters (*O1*, *Len1*) and (*O2*, *Len2*), the correlator block is applied twice, one time for each set, and the corresponding output values (the sums and the inner products) are summed.

DC removal

The inner products computed by the correlator are modified as follows:

$X^TX = X^TX - (\Sigma X)^2 / LW$

$Y^TY = Y^TY - (\Sigma Y)^2 / LW$

$Z^TZ = Z^TZ - (\Sigma Z)^2 / LW$

$X^TY = X^TY - \Sigma X \times \Sigma Y/LW$

$X^TZ = X^TZ - \Sigma X \times \Sigma Z/LW$

$Y^TZ = Y^TZ - \Sigma Y \times \Sigma Z/LW$

Interpolation

Correlation score *CS* is computed by the following interpolation formula:

$$CS = \frac{\beta \times X^T Z + \alpha \times Y^T Z}{\sqrt{Z^T Z \times \left(\beta^2 \times X^T X + 2\alpha\beta \times X^T Y + \alpha^2 \times Y^T Y\right)}}$$

where:

$$\alpha = i\tau - \tau,\ \beta = 1 - \alpha$$

Finally, *CS* value is truncated if it falls outside the interval [0, 1].

$$CS = \max(CS,0), \quad CS = \min(CS,1).$$

4.2.15.7 Pitch estimate selection

Input to this stage is the set of pitch candidates. Each candidate ($F0_k$, SS_k, CS_k) is represented by the corresponding pitch frequency $F0_k$, spectral score (the utility function value) SS_k and correlation score CS_k. The block outputs a pitch estimate (F0, SS, CS) which either is selected among the candidates or indicates that that the frame represents unvoiced speech in which case F0 is set to 0.

Pitch estimate selection block might be entered several (at most 3) times during the processing of one frame. It is entered after pitch candidates generation is performed for each pitch search interval SRi. Each time the list of pitch candidates which is fed into the block is updated appropriately to include all the pitch candidates detected so far. Thus the list passed into this block after the processing of SR3 search range includes the candidates found within this range, typically two candidates. If one of the candidates is selected as the pitch estimate then the pitch estimation process terminates and the control flows to the history information update block (described below in clause 4.2.15.8). Otherwise the candidates generated within the SR2 range are combined with the ones found within SR3 and the combined list (typically containing four candidates) is fed into pitch estimate selection block. If no pitch estimate is selected at this time the block is entered again after SR1 range is processed. At this time the candidate list contains the candidates generated in all the three ranges (typically 6 candidates). A variable *EPT* which is fed to the block along with the candidates list indicates whether the list contains candidates generated for all the three search ranges (*EPT = 1*) or not (*EPT = 0*).

The selection process is shown on the flow-chart of figure 4.4.

The candidates are sorted at step 100 in descending order of their *F0* values. Then at step 110 the candidates are scanned sequentially until a candidate of *class 1* is found, or all the candidates are tested. A candidate is defined to be of class 1 if the *CS* and *SS* values associated with the candidate satisfy the following condition:

(CS ≥ C1 AND SS ≥ S1) OR (SS ≥S11 AND SS + CS ≥ CS1) (Class 1 condition)

where:

C1 = 0,79; S1 = 0,78; S11 = 0,68 and CS1 = 1,6

At step 130 the flow branches. If a class 1 candidate is found it is selected to be a *preferred candidate*, and the control is passed to step 140 performing a *Find Best in Vicinity* procedure described by the following. Those candidates among the ones following in the list the preferred candidate are checked to determine those ones which are close in terms of *F0* to the preferred candidate. Two values $F0_1$ and $F0_2$ are defined to be close to each other if:

($F0_1$ < 1,2 × $F0_2$ AND $F0_2$ < 1,2 × $F0_1$) (Closeness condition).

A plurality of *better* candidates is determined among the close candidates. A better candidate must have a higher SS and a higher CS values than those of the preferred candidate respectively. If at least one better candidate exists then the *best* candidate is determined among the better candidates. The best candidate is characterized by that there is no other better candidate, which has a higher SS and a higher CS values than those of the best candidate respectively. The best candidate is selected to be a preferred candidate instead of the former one. If no better candidate is found the preferred candidate remains the same.

At step 150 the candidates following the preferred candidate are scanned one by one until either a candidate of class 1 is found whose scores $SS_{candidate}$ and $CS_{candidate}$ satisfy following condition:

$$SS_{candidate} + CS_{candidate} \geq SS_{preferred} + CS_{preferred} + 0,18$$

or all the candidates are scanned. If a candidate is found which meets the above condition it is selected to be the preferred candidate and Find Best in Vicinity procedure is applied. Otherwise the control is passed directly to step 180, where the *EPT* variable value is tested. If *EPT* indicates that all the pitch search ranges have been processed the pitch estimate is set to the preferred candidate. Otherwise the following condition is tested:

$$SS_{preferred} \geq 0,95 \text{ AND } CS_{preferred} \geq 0,95$$

If the condition is satisfied the pitch estimate is set to the preferred candidate, otherwise the pitch frequency *F0* is set to 0 indicating that no pitch is detected.

Returning to the conditional branching step 130, if no class 1 candidate is found then at step 120 it is checked if the *StableTrackF0* variable has non-zero value in which case the control is passed to step 210, otherwise step 270 is performed.

At step 210 a reference fundamental frequency value $F0_{ref}$ is set to *StableTrackF0*. Then at step 220 the candidates are scanned sequentially until either a candidate of a *class 2* is found or all the candidates are tested. A candidate is defined to be of class 2 if the frequency and the score values associated with it satisfy the condition:

$$(CS > C2 \text{ AND } SS > S2) \text{ AND } (1/1,22 < |F0/F0_{ref}| < 1,22) \qquad \textit{(Class 2 condition)}$$

where:

$$C2 = 0,7;\ S2 = 0,7$$

If no class 2 candidate is found then the pitch estimate is set to 0 at step 240. Otherwise, the class 2 candidate is chosen to be the preferred candidate and Find Best in Vicinity procedure is applied at step 250. Then at step 260 the pitch estimate is set to the preferred candidate.

Returning to the conditional branching step 120, if $StableTrackF0 = 0$ then control is passed to step 270 where a *Continuous Pitch Condition*:

$$PrevF0 > 0 \text{ AND } StablePitchCount > 1$$

is tested. (StablePitchCount variable is described below in clause 4.2.15.8.) If the condition is satisfied then at step 280 the frequency reference value $F0_{ref}$ is set to *PrevF0* and the class 2 candidate search is performed at step 290. If a class 2 candidate is found (test step 300) then it is selected as the preferred candidate, Find Best In Vicinity procedure is applied at step 310, and the pitch estimate is set to the preferred candidate at step 320. Otherwise, the processing proceeds with step 330 likewise it happens if Continuous Pitch Condition test of step 270 fails.

At step 330 the candidates are scanned sequentially until a candidate of *class 3* is found or all the candidates are tested. A candidate is defined to be of class 3 if the scores associated with it satisfy the condition:

$$(CS \geq C3 \text{ OR } SS \geq S3) \qquad \textit{(Class 3 condition)}$$

where:

$$C3 = 0,85;\ S3 = 0,82$$

If no class 3 candidate is found then the pitch frequency is set to 0. Otherwise, the class 3 candidate is selected as the preferred candidate, and Find Best in Vicinity procedure is applied at step 360. Then at step 370 the pitch estimate is set to the preferred candidate.

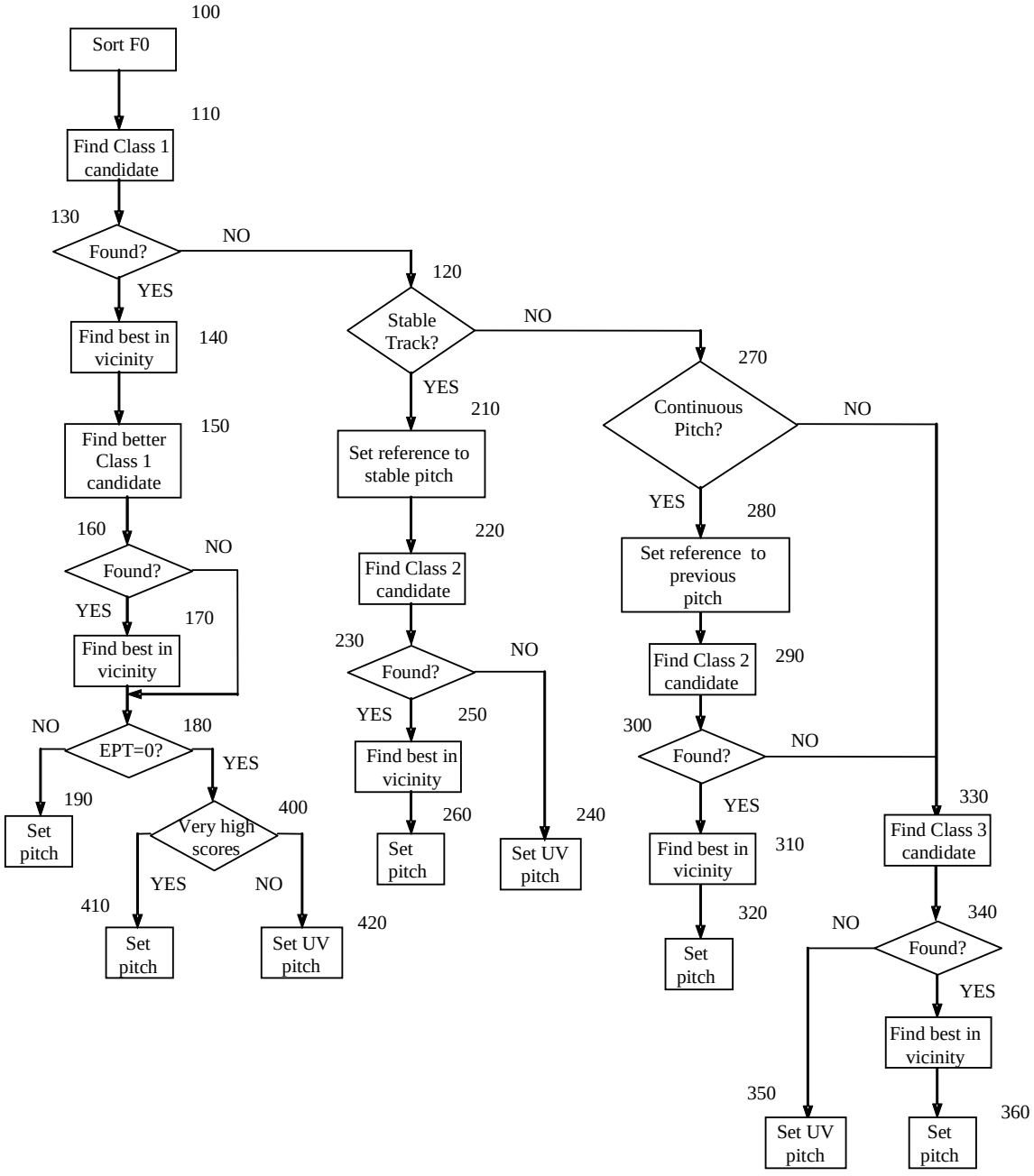


Figure 4.4: Pitch estimate selection

4.2.15.8 History information update

The pitch estimator maintains following variables holding information on the estimation process history: *PrevF0, StableTrackF0, StablePitchCount* and *DistFromStableTrack.*

The variables are initialized as follows:

PrevF0 = 0, StablePitchCount = 0, DistFromStableTrack = 1 000, StableTrackF0 = 0.

The variables are updated at each frame after pitch estimation processing is completed and the pitch frequency estimate *F0* is set. The update process is described by the following pseudo code section.

```
if (F0 > 0  AND  PrevF0 > 0  AND 1/1.22 < |F0/PrevF0| < 1.22)
    StablePitchCount = StablePitchCount + 1;
else
    StablePitchCount = 0;

if (StablePitchCount ≥ 6)
{
    DistFromStableTrack = 0;
    StableTrackF0 = F0;
}
else if (DistFromStableTrack ≤ 2)
{
    if (StableTrackF0 > 0  AND 1/1.22 < |F0/StableTrackF0| < 1.22)
    {
        DistFromStableTrack = 0;
        StableTrackF0 = F0;
    }
    else
        DistFromStableTrack = DistFromStableTrack + 1;
}
else {
    StableTrackF0 = 0;
    DistFromStableTrack = DistFromStableTrack + 1;
}
PrevF0 = F0;
```

4.2.15.9 Output pitch value

The pitch frequency estimate *F0* is converted to an output pitch value *P* representing pitch period duration measured in sampling intervals corresponding to 8 kHz sampling rate.

$$P = \begin{cases} 0 \ if \ \ F0 = 0 \\ 8\,000 / F0 \ otherwise \end{cases}$$

4.2.16 Classification

The inputs to the classification block are the *vad_flag* and *hangover_flag* from the *VAD* block, the frame energy *E* from the *EC* block, the offset-free input signal s_{of} from the *Offcom* block, the upper-band signal s_{ub} from the *PP* block, and the pitch period estimate *P* from the *PITCH* block. The output of the classification block is the voicing class *VC*, which is one of the output parameters of the front-end.

The voicing class *VC* is estimated from the different inputs to the classification block as follows. From the upper-band signal s_{ub} and the frame energy *E*, the upper-band energy fraction EF_{ub} is computed as:

$$EF_{ub} = \frac{\sum_{i=1}^{N} s_{ub}(i)^2}{E} \tag{4.51}$$

From the offset-free input signal s_{of}, the zero-crossing measure *ZCM* is computed as follows.

$$ZCM = \frac{1}{2(N-1)} \sum_{i=2}^{N} |\,\mathrm{sgn}[s_{of}(i)] - \mathrm{sgn}[s_{of}(i-1)]\,| \tag{4.52}$$

where:

$$\mathrm{sgn}[s_{of}(i)] = \begin{cases} +1, \ s_{of}(i) \geq 0 \\ -1, \ s_{of}(i) < 0 \end{cases} \tag{4.53}$$

The logic used by the classification block is illustrated by the pseudo-code below.

```
if (vad_flag == FALSE)
    VC = "non-speech";
else if (P == 0)
    VC = "unvoiced";
else if ((hangover_flag == TRUE) || (EF_ub ≤ EF_UB_THLD) || (ZCM >= ZCM_THLD))
    VC = "mixed-voiced";
else
    VC = "fully-voiced";
end
```

The upper-band energy fraction threshold EF_UB_THLD is 0,0018 for 8 kHz, 0,0023 for 11 kHz, and 0,0029 for 16 kHz sampling frequency respectively. The zero-crossing measure threshold ZCM_THLD is 0,4375.

4.2.17 Front-end output

The final feature vector consists of 16 coefficients: the log-energy coefficient (clause 4.2.5), the 13 cepstral coefficients (clause 4.2.11), the pitch period (clause 4.2.14), and the voicing class (clause 4.2.15).

The C_0 coefficient is often redundant when the log-energy coefficient is used. However, the feature extraction algorithm is defined here for both energy and C_0. Depending on the application, either the C_0 coefficient or the log-energy coefficient may be used.

5 Feature compression algorithm

5.1 Introduction

This clause describes the distributed speech recognition front-end feature vector compression algorithm. The algorithm makes use of the parameters from the front-end feature extraction algorithm of clause 4. Its purpose is to reduce the number of bits needed to represent each front-end feature vector.

5.2 Compression algorithm description

5.2.1 Input

The compression algorithm is designed to take the feature parameters for each short-time analysis frame of speech data as they are available and as specified in clause 4.

Fourteen of the sixteen parameters are compressed using a Vector Quantizer (VQ). The input parameters for the VQ are the first twelve static mel cepstral coefficients:

$$\mathbf{c}(m) = [c_1(m), c_2(m) \ldots c_{12}(m)]^T \tag{5.1}$$

where m denotes the frame number, plus the zeroth cepstral coefficient and a log energy term as defined in clause 4.2.16. These parameters are formatted as:

$$y(m) = \begin{bmatrix} \mathbf{c}(m) \\ c_0(m) \\ \log[E(m)] \end{bmatrix} \tag{5.2}$$

The remaining two parameters, viz., pitch period and class, are compressed jointly using absolute and differential scalar quantization techniques.

5.2.2 Vector quantization

The feature vector ***y(m)*** is directly quantized with a split vector quantizer. Coefficients are grouped into pairs, and each pair is quantized using its own VQ codebook. The resulting set of index values is then used to represent the corresponding speech parameters. Coefficient pairings (by front-end parameter) are shown in table 5.1, along with the codebook size used for each pair.

The closest VQ centroid is found using a weighted Euclidean distance to determine the index:

$$d_j^{i,i+1} = \begin{bmatrix} y_i(m) \\ y_{i+1}(m) \end{bmatrix} - q_j^{i,i+1} \tag{5.3}$$

$$idx^{i,i+1}(m) = \underset{0 \le j \le (N^{i,i+1}-1)}{\operatorname{argmin}} \left\{ \left(d_j^{i,i+1}\right) W^{i,i+1} \left(d_j^{i,i+1}\right) \right\}, \quad i = 0,2,4 \ldots 12 \tag{5.4}$$

where $q_j^{i,i+1}$ denotes the *j*th codevector in the codebook $Q^{i,i+1}$, $N^{i,i+1}$ is the size of the codebook, $W^{i,i+1}$ is the (possibly identity) weight matrix to be applied for the codebook $Q^{i,i+1}$, and $idx^{i,i+1}(m)$ denotes the codebook index chosen to represent the vector $[y_i(m), y_{i+1}(m)]^T$. The indices are then retained for transmission to the back-end.

Table 5.1: Split vector quantization feature pairings

	Size	Weight Matrix		
Codebook	**($N^{I,I+1}$)**	**($W^{I,I+1}$)**	**Element 1**	**Element 2**
$Q^{0,1}$	64	I	c_1	c_2
$Q^{2,3}$	64	I	c_3	c_4
$Q^{4,5}$	64	I	c_5	c_6
$Q^{6,7}$	64	I	c_7	c_8
$Q^{8,9}$	64	I	c_9	c_{10}
$Q^{10,11}$	64	I	c_{11}	c_{12}
$Q^{12,13}$	256	Non-identity	c_0	log[E]

Two sets of VQ codebooks are defined; one is used for speech sampled at 8 kHz or 11 kHz while the other for speech sampled at 16 kHz. The numeric values of these codebooks and weights are specified as part of the software implementing the standard. The weights used (to one decimal place of numeric accuracy) are:

8 kHz or 11 kHz sampling rate $\quad W^{12,13} = \begin{bmatrix} 1\,446{,}0 & 0 \\ 0 & 14{,}7 \end{bmatrix}$

16 kHz sampling rate $\quad W^{12,13} = \begin{bmatrix} 1\,248{,}9 & 0 \\ 0 & 12{,}7 \end{bmatrix}$

5.2.3 Pitch and class quantization

The pitch period of a frame can range from 19 samples to 140 samples (both inclusive) at 8 kHz sampling rate. The voicing class of a frame can be one of the following four:

- *non-speech;*
- *unvoiced speech;*
- *mixed-voiced speech;* and
- *(fully) voiced speech.*

The class information of a frame is represented jointly using the pitch and class indices. The pitch information of alternate frames is quantized absolutely using 7 bits or differentially using 5 bits.

5.2.3.1 Class quantization

When the voicing class of a frame is non-speech or unvoiced speech, the pitch index of the corresponding frame is chosen to be *zero*, i.e. all-zero codeword either 5 bits or 7 bits long. For non-speech, the 1-bit class index is chosen as 0, and for unvoiced speech, the class index is chosen as 1. For such frames, the pitch period is indeterminate.

When the voicing class of a frame is mixed-voiced speech or (fully) voiced speech, the pitch index of the corresponding frame is chosen to be some index other than zero, either 5 bits or 7 bits long. For mixed-voiced speech, the 1-bit class index is chosen as 0, and for (fully) voiced speech, the class index is chosen as 1. For such frames, the pitch index specifies the pitch period as discussed under clause 5.2.3.2.

Thus the pitch and class indices of a frame jointly determine the voicing class of the frame as illustrated in table 5.2.

Table 5.2: Class quantization

Voicing Class (VC)	Pitch index (Pidx)	Class index (Cidx)
Non-speech	0	0
Unvoiced-speech	0	1
Mixed-voiced speech	> 0	0
Fully-voiced speech	> 0	1

5.2.3.2 Pitch quantization

The pitch period of an even-numbered frame (with the starting frame numbered zero), or equivalently, the first frame of each frame pair is quantized absolutely using 7 bits. Out of the 128 indices ranging from 0 to 127, the index 0 is reserved for indicating that the voicing class is non-speech or unvoiced speech as discussed under clause 5.2.3.1. The remaining 127 indices are assigned in increasing order to 127 quantization levels that span the range from 19 to 140 uniformly in the log-domain. Given the pitch period of the frame, the quantization level that is closest to the pitch period in the Euclidean sense and the corresponding index are chosen.

$$Pidx(m) = \underset{1 \le j \le 127}{\arg\min} (P(m) - q_j)^2 \qquad (5.5)$$

where $P(m)$ is the pitch period of the m^{th} frame (m even), q_j is the j^{th} quantization level, and $Pidx(m)$ is the pitch quantization index for the m^{th} frame.

The pitch period of an odd-numbered frame (with the starting frame numbered zero), or equivalently, the second frame of each frame pair is quantized differentially using 5 bits. Out of the 32 indices ranging from 0 to 31, the index 0 is reserved for indicating that the voicing class is non-speech or unvoiced speech as discussed under clause 5.2.3.1. The remaining 31 indices are assigned in increasing order to 31 quantization levels, which are chosen depending on which of the three preceding quantized pitch periods serves as the reference (for differential quantization) and what its value is. The choice of the reference pitch period and the 31 quantization levels for different situations are illustrated in table 5.3. With reference to the table, a quantized pitch period value with a non-zero index may be *reliable* or *unreliable* to serve as a reference. An absolutely quantized pitch period value is always considered reliable. A differentially quantized pitch period value is considered reliable only if the reference value used for its quantization is the quantized pitch period value of the preceding frame. In the table, the different quantization levels are specified as a factor that multiplies the chosen reference value. If any quantization level falls outside the pitch range of 19 to 140, then it is limited to the appropriate boundary value.

Table 5.3: Choice of reference and quantization levels for differential quantization

Pitch indices of preceding 3 frames			Choice of reference pitch period and 31 quantization levels for $(m+1)^{th}$ frame
Pidx(m-2)	Pidx(m-1)	Pidx(m)	
0	0 OR > 0 but unreliable	0	No suitable reference is available. Use 5-bit absolute quantization. The 31 quantization levels are chosen to span the range from 19 to 140 uniformly in the log-domain.
Do not care	Do not care	> 0	The quantized pitch period value of the m^{th} frame is chosen as the reference. Out of the 31 quantization levels, 27 are chosen to cover the range from (0,8163 × reference) to (1,2250 × reference) uniformly in the log-domain. The other 4 levels depend on the reference value as follows: 19 ≤ reference ≤ 30 - (2,00, 3,00, 4,00, 5,00) × reference 30 < reference ≤ 60 - (1,50, 2,00, 2,50, 3,00) × reference 60 < reference ≤ 95 - (0,50, 0,67, 1,50, 2,00) × reference 95 < reference ≤ 140 - (0,25, 0,33, 0,50, 0,67) × reference
Do not care	> 0 Reliable	0	The quantized pitch period value of the $(m-1)^{th}$ frame is chosen as the reference. The choice of quantization levels is the same as shown in the row below.
> 0	0 OR > 0 but unreliable	0	The quantized pitch period value of the $(m-2)^{th}$ frame is chosen as the reference. Out of the 31 quantization levels, 25 are chosen to cover the range from (0,7781 × reference) to (1,2852 × reference) uniformly in the log-domain. The other 6 levels depend on the reference value as follows: 19 ≤ reference ≤ 30 - (1,50, 2,00, 2,50, 3,00, 4,00, 5,00) × reference 30 < reference ≤ 60 - (0,67, 1,50, 2,00, 2,50, 3,00, 4,00) × reference 60 < reference ≤ 95 - (0,33, 0,50, 0,67, 1,50, 1,75, 2,00) × reference 95 < reference ≤ 140 - (0,20, 0,25, 0,33, 0,50, 0,67, 1,50) × reference

The 31 indices used for differential quantization are assigned in increasing order to the 31 quantization levels. Given the pitch period of the frame, the quantization level that is closest to the pitch period in the Euclidean sense and the corresponding index are chosen.

$$Pidx(m+1) = \underset{1 \le j \le 31}{\arg\min} (P(m+1) - q_j)^2 \qquad (5.6)$$

where $P(m+1)$ is the pitch period of the $(m+1)^{th}$ frame (m even), q_j is the j^{th} quantization level, and $Pidx(m+1)$ is the pitch quantization index for the $(m+1)^{th}$ frame.

6 Framing, bit-stream formatting, and error protection

6.1 Introduction

This clause describes the format of the bitstream used to transmit the compressed feature vectors. The frame structure used and the error protection that is applied to the bitstream is defined.

6.2 Algorithm description

6.2.1 Multiframe format

In order to reduce the transmission overhead, each multiframe message packages speech features from multiple short-time analysis frames. A multiframe, as shown in table 6.1, consists of a synchronization sequence, a header field, and a stream of frame packets.

Table 6.1: Multiframe format

Sync Sequence	Header Field	Frame Packet Stream
<- 2 octets ->	<- 4 octets ->	<- 162 octets ->
<- 168 octets ->		

In order to improve the error robustness of the protocol, the multiframe has a fixed length (168 octets). A multiframe represents 240 ms of speech, resulting in a data rate of 5 600 bit/s.

In the specification that follows, octets are transmitted in ascending numerical order; inside an octet, bit 1 is the first bit to be transmitted. When a field is contained within a single octet, the lowest-numbered bit of the field represents the lowest-order value (or the least significant bit). When a field spans more than one octet, the lowest-numbered bit in the first octet represents the lowest-order value (LSB), and the highest-numbered bit in the last octet represents the highest-order value (MSB). An exception to this field mapping convention is made for the Cyclic Redundancy Code (CRC) fields. For these fields, the lowest numbered bit of the octet is the highest-order term of the polynomial representing the field. In simple stream formatting diagrams (e.g. table 6.1) fields are transmitted left to right.

6.2.2 Synchronization sequence

Each multiframe begins with the 16-bit synchronization sequence 0 × 87B2 (sent LSB first, as shown in table 6.2).

The inverse synchronization sequence 0 × 784D can be used for synchronous channels requiring rate adaptation. Each multiframe may be preceded or followed by one or more inverse synchronization sequences. The inverse sync is not required if a multiframe is immediately followed by the sync sequence for the next multiframe.

Table 6.2: Multiframe synchronization sequence

Bit	8	7	6	5	4	3	2	1	Octet
	1	0	0	0	0	1	1	1	1
	1	0	1	1	0	0	1	0	2

6.2.3 Header field

Following the synchronization sequence, a header field is transmitted. Due to the critical nature of the data in this field, it is represented in a (31, 16) extended systematic codeword. This code will support 16-bits of data and has an error correction capability for up to three bit errors, an error detection capability for up to seven bit errors, or a combination of both error detection and correction.

Ordering of the message data and parity bits is shown in table 6.3, and definition of the fields appears in table 6.4. The 4 bit multiframe counter gives each multiframe a modulo-16 index. The counter value for the first multiframe is "0001". The multiframe counter is incremented by one for each successive multiframe until the final multiframe. The final multiframe is indicated by zeros in the frame packet stream (see clause 6.2.4).

NOTE: The remaining eight bits which are currently undefined are left for future expansion. A fixed length field has been chosen for the header in order to improve error robustness and mitigation capability.

Table 6.3: Header field format

Bit	8	7	6	5	4	3	2	1	Octet
	Ext	MframeCnt				FeType	SampRate		1
	EXP8	EXP7	EXP6	EXP5	EXP4	EXP3	EXP2	EXP1	2
	P8	P7	P6	P5	P4	P3	P2	P1	3
	P16	P15	P14	P13	P12	P11	P10	P9	4

Table 6.4: Header field definitions

Field	No. Bits	Meaning	Code	Indicator
SampRate	2	sampling rate	00	8 kHz
			01	11 kHz
			10	undefined
			11	16 kHz
FeType	1	Front-end specification	0	standard
			1	Noise robust
MframeCnt	4	multiframe counter	xxxx	Modulo-16 number
Ext	1	Extended front-end	0	Not extended (4 800 bps)
			1	Extended (5 600 bps)
EXP2 - EXP9	8	Expansion bits (TBD)	0	(zero pad)
P1 - P16	16	Cyclic code parity bits	(see below)	

The generator polynomial used is:

$$g_1(X) = 1 + X^8 + X^{12} + X^{14} + X^{15} \tag{6.1}$$

The proposed (31, 16) code is extended, with the addition of an (even) overall parity check bit, to 32 bits. The parity bits of the codeword are generated using the calculation.

$$\begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \\ P_5 \\ P_6 \\ P_7 \\ P_8 \\ P_9 \\ P_{10} \\ P_{11} \\ P_{12} \\ P_{13} \\ P_{14} \\ P_{15} \\ P_{16} \end{bmatrix} = \begin{bmatrix} 1&0&0&0&0&0&0&0&1&0&0&0&1&0&1&1 \\ 1&1&0&0&0&0&0&0&1&1&0&0&1&1&1&0 \\ 1&1&1&0&0&0&0&0&1&1&1&0&1&1&0&1 \\ 0&1&1&1&0&0&0&0&0&1&1&1&0&1&1&1 \\ 1&0&1&1&1&0&0&0&1&0&1&1&0&0&0&0 \\ 0&1&0&1&1&1&0&0&0&1&0&1&1&0&0&0 \\ 0&0&1&0&1&1&1&0&0&0&1&0&1&1&0&0 \\ 0&0&0&1&0&1&1&1&0&0&0&1&0&1&1&0 \\ 1&0&0&0&1&0&1&1&0&0&0&0&0&0&0&1 \\ 0&1&0&0&0&1&0&1&1&0&0&0&0&0&0&1 \\ 0&0&1&0&0&0&1&0&1&1&0&0&0&0&0&1 \\ 0&0&0&1&0&0&0&1&0&1&1&0&0&0&0&1 \\ 0&0&0&0&1&0&0&0&1&0&1&1&0&0&0&1 \\ 0&0&0&0&0&1&0&0&0&1&0&1&1&0&0&1 \\ 0&0&0&0&0&0&1&0&0&0&1&0&1&1&0&1 \\ 0&0&0&0&0&0&0&1&0&0&0&1&0&1&1&1 \end{bmatrix}^T \times \begin{bmatrix} SampRate1 \\ SampRate2 \\ feType \\ MFrameCnt1 \\ MFrameCnt2 \\ MFrameCnt3 \\ MFrameCnt4 \\ Ext \\ EXP1 \\ EXP2 \\ EXP3 \\ EXP4 \\ EXP5 \\ EXP6 \\ EXP7 \\ EXP8 \end{bmatrix} \tag{6.2}$$

Where T denotes the matrix transpose.

6.2.4 Frame Packet Stream

Each 10 ms frame from the front-end is represented by the codebook indices specified in clause 5.2.2 as well as the pitch index and class index specified in clause 5.2.3. The indices for a pair of frames are formatted according to table 6.5.

NOTE: The exact alignment with octet boundaries will vary from frame pair to frame pair.

Table 6.5: Frame information for m^{th} and $(m+1)^{th}$ frames

Bit	8	7	6	5	4	3	2	1	Octet
	$Idx^{2,3}(m)$		$Idx^{0,1}(m)$						1
	$Idx^{4,5}(m)$				$Idx^{2,3}(m)$ (cont)				2
	$Idx^{6,7}(m)$						$Idx^{4,5}(m)$ (cont)		3
	$Idx^{10,11}(m)$		$Idx^{8,9}(m)$						4
	$Idx^{12,13}(m)$				$Idx^{10,11}(m)$ (cont)				5
	$Idx^{0,1}(m+1)$				$Idx^{12,13}(m)$ (cont)				6
	$Idx^{2,3}(m+1)$						$Idx^{0,1}(m+1)$ (cont)		7
	$Idx^{6,7}(m+1)$		$Idx^{4,5}(m+1)$						8
	$Idx^{8,9}(m+1)$				$Idx^{6,7}(m+1)$ (cont)				9
	$Idx^{10,11}(m+1)$						$Idx^{8,9}(m+1)$ (cont)		10
	$Idx^{12,13}(m)$								11
	Pidx(m)				CRC(m,m+1)				12
	Pidx(m+1)					Pidx(m) (cont)			13
					PC-CRC(m,m+1)		Cidx(m+1)	Cidx(m)	14

The codebook indices for each frame take up 44 bits. After two frames worth of codebook indices, or 88 bits, a 4-bit CRC ($g(X) = 1 + X + X^4$) calculated on these 88 bits immediately follows it. The pitch indices of the first frame (7 bits) and the second frame (5 bits) of the frame pair then follow. The class indices of the two frames in the frame pair worth 1 bit each next follow. Finally, a 2-bit CRC (denoted by PC-CRC) calculated on the pitch and class bits (total: 14 bits) of the frame pair using the binary polynomial $g(X) = 1 + X + X^2$ is included. The total number of bits in frame pair packet is therefore 44 + 44 + 4 + 7 + 5 + 1 + 1 + 2 = 108 octets or 13,5 octets. Twelve of these frame pair packets are combined to fill the 162 octet (1 296 bit) feature stream. When the feature stream is combined with the overhead of the synchronization sequence and the header, the resulting format requires a data rate of 5 600 bit/s.

All trailing frames within a final multiframe that contain no valid speech data will be set to all zeros.

7 Bit-stream decoding and error mitigation

7.1 Introduction

This clause describes the algorithms used to decode the received bitstream to regenerate the speech feature vectors. It also covers the error mitigation algorithms that are used to minimize the consequences of transmission errors on the performance of a speech recognizer and/or a back-end speech reconstructor.

7.2 Algorithm description

7.2.1 Synchronization sequence detection

The method used to achieve synchronization is not specified in the present document. The detection of the start of a multiframe may be done by the correlation of the incoming bit stream with the synchronization flag. The output of the correlator may be compared with a correlation threshold (the value of which is not specified in this definition). Whenever the output is equal to or greater than the threshold, the receiver should decide that a flag has been detected. For increased reliability in the presence of errors the header field may also be used to assist the synchronization method.

7.2.2 Header decoding

The decoder used for the header field is not specified in the present document. When the channel can be guaranteed to be error-free, the systematic codeword structure allows for simple extraction of the message bits from the codeword. In the presence of errors, the code may be used to provide either error correction, error detection, or a combination of both moderate error correction capability and error detection capability.

In the presence of errors, the decoding of the frame packet stream in a multiframe is not started until at least two headers have been received in agreement with each other. Multiframes are buffered for decoding until this has occurred. The header block in each received multiframe has its cyclic error correction code decoded and the "common information carrying bits" are extracted. With the header defined in the present document the "common information carrying bits" consist of SampRate, FeType, Ext, and EXP1- EXP8 (expansion bits).

NOTE: The use of EXP1 - EXP8 depends on the type of information they may carry in the future. Only those bits which do not change between each multiframe are used in the check of agreement described above.

Once the common information carrying bits have been determined then these are used for all the multiframes in a contiguous sequence of multiframes.

7.2.3 Feature decompression

Codebook, pitch, and class indices are extracted from the frame packet stream, with optional checking of CRC and PC-CRC (back-end handling of frames failing the CRC and PC-CRC check is specified in clause 7.2.4). Using the codebook indices received, estimates of the mel-cepstrum and log*E* features are extracted with a VQ codebook lookup.

$$\begin{bmatrix} \hat{y}_i(m) \\ \hat{y}_{i+1}(m) \end{bmatrix} = Q^{i,i+1}_{idx^{i,i+1}(m)} \quad i = 0, 2, 4 \ldots 12 \tag{7.1}$$

From the pitch and class indices, the voicing class feature is extracted as specified in table 5.2. For non-speech and unvoiced frames, the pitch period is indeterminate. For a mixed-voiced or (fully) voiced frame, the pitch period is estimated from the pitch index as follows. For a frame with absolute pitch quantization (*m* even), the pitch index directly specifies the quantized pitch period. For a frame with differential pitch quantization (*m* odd), the pitch index specifies the factor by which the reference has to be multiplied. The reference, which can be the quantized pitch period value of any one of the preceding three frames, is obtained using the rules of table 5.3. If no suitable reference is available (Row 1 of table 5.3), then the pitch index directly specifies the quantized pitch period.

7.2.4 Error mitigation

7.2.4.1 Detection of frames received with errors

When transmitted over an error prone channel, the received bitstream may contain errors. Two methods are used to determine if a frame pair packet has been received with errors:

- CRC and PC-CRC: The CRC recomputed from the codebook indices of the received frame pair packet data does not match the received CRC for the frame pair, or, the PC-CRC recomputed from the pitch and class indices of the received frame pair packet data does not match the received PC-CRC for the frame pair, or both.
- Data consistency: A heuristic algorithm to determine whether or not the decoded parameters for each of the two speech vectors in a frame pair packet are consistent. The details of this algorithm are described below.

The parameters corresponding to each codebook index, $idx^{i, i+1}$, of the two frames within a frame packet pair are compared to determine if either of the indices are likely to have been received with errors:

$$badindexflag_i = \begin{cases} 1 & if \left(y_i(m+1) - y_i(m) \rangle T_i \right) OR \left(y_{i+1}(m+1) - y_{i+1}(m) \rangle T_{i+1} \right) \\ 0 & otherwise \end{cases} \quad i = 0, 2, \ldots 12 \tag{7.2}$$

The thresholds T_i have been determined based on measurements on unerrored speech. A voting algorithm is applied to determine if the whole frame pair packet is to be treated as if it had been received with transmission errors. The frame pair packet is classified as received with error if:

$$\sum_{i=0,2,...12} badindexflag_i \geq 2 \qquad (7.3)$$

The data consistency check for errored data is only applied when frame pair packets failing the CRC test are detected. It is applied to the frame pair packet received before the one failing the CRC test and successively to frames after one failing the CRC test until one is found that passes the data consistency test. The details of this algorithm are shown in the flow charts of figures 7.1 and 7.2.

7.2.4.2 Substitution of parameter values for frames received with errors

The parameters from the last speech vector received without errors before a sequence of one or more "bad" frame pair packets and those from the first good speech vector received without errors afterwards are used to determine replacement vectors to substitute for those received with errors. If there are B consecutive bad frame pairs (corresponding to 2B speech vectors) then the first B speech vectors are replaced by a copy of the last good speech vector before the error and the last B speech vectors are replaced by a copy of the first good speech vector received after the error.

In the presence of errors, the decoding of the frame packet stream in a multiframe is not started until at least two headers have been received in agreement with each other. Multiframes are buffered for decoding.

7.2.4.3 Modification of parameter values for frames received with errors

The log*E*, pitch, and class parameters of frames received with errors are modified as follows after the substitution step described in clause 7.2.4.2. This modification step affects only back-end speech reconstruction - it does not affect speech recognition.

First, a 3-point median filter is applied to the log*E* parameter. The median value of the log*E* parameters of the preceding, current, and succeeding frames replaces the log*E* parameter of the current frame. The median filter is switched on only after the first frame error has been detected. In other words, there is no median filtering for an error-free channel.

Second, the log*E*, pitch, and class parameters of frames received with errors are modified according to the runlength of errors. Let the runlength of errors be 2B frames. If 2B is less than or equal to 4, no parameter modification is done. In this case, because of the substitution step in clause 7.2.4.2, the first B frames receive their parameters from the good frame on the left (before the error) and the next B frames receive their parameters from the good frame on the right (after the error).

For a runlength greater than 4 but less than or equal to 24, the parameter modification is done as follows. The parameters of the first two frames and last two frames are not modified. From the 3rd frame to the Bth frame, the log*E* parameter is decreased linearly from left to right by 2 per frame. The value of the log*E* parameter is however not allowed to go below 4,7. If these frames are (fully) voiced, then they are modified to mixed-voiced frames. The pitch parameters are not changed. From the (2B-2)th frame to (B+1)th frame (both inclusive), the log*E* parameter is decreased linearly from right to left by 2 per frame with a floor value of 4,7. Fully voiced frames are modified to mixed-voiced frames and the pitch parameters are not modified.

If the runlength of errors is greater than 24, then the first 12 and the last 12 frames are handled exactly as above. The remaining (2B-12) frames in the middle are modified as follows. The log*E* parameter is set to 4,7, the class parameter is set to "unvoiced", and the pitch parameter is indeterminate.

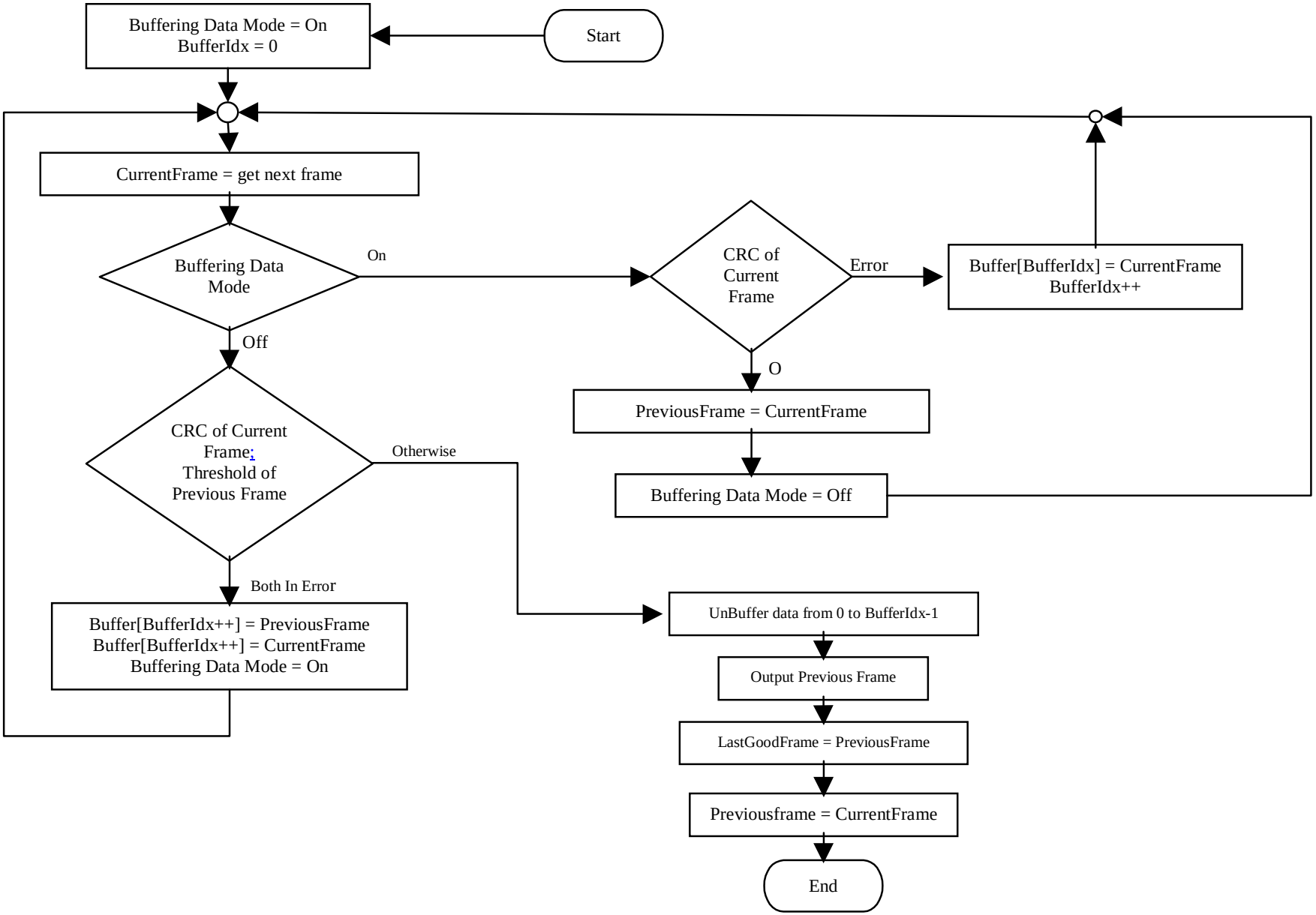


Figure 7.1: Error mitigation initialization flow chart

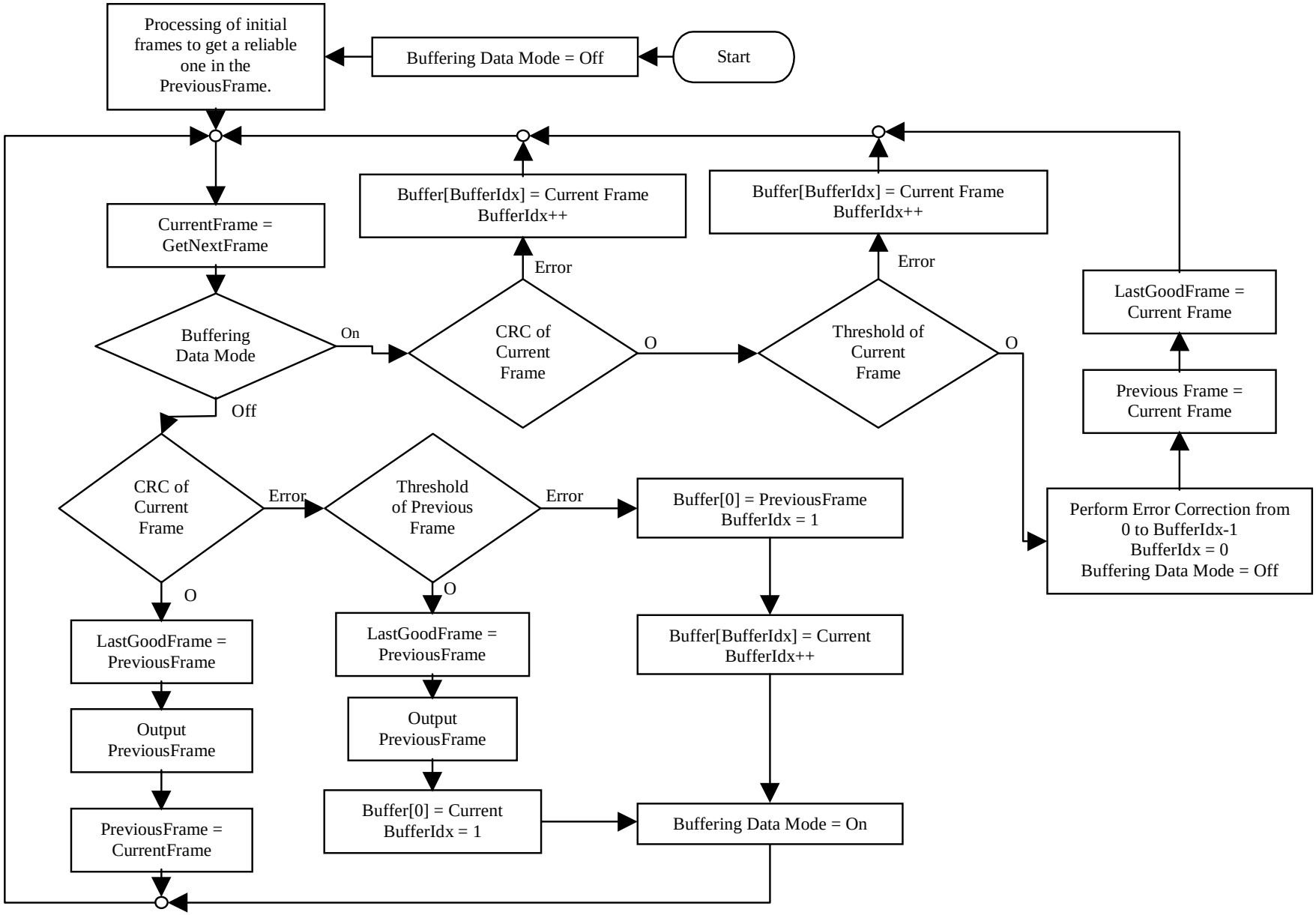


Figure 7.2: Main error mitigation flow chart

8 Server side speech reconstruction

8.1 Introduction

This clause describes the server side speech reconstruction algorithm. Speech is reconstructed from feature vectors that have been decoded from the received bit stream and error-mitigated. Each feature vector consists of the following 16 parameters - 13 Mel-Frequency Cepstral Coefficients (MFCC) C_0 through C_{12}, the log-energy parameter log$E$, the pitch period value P, and the voicing class VC. The reconstructed speech is in digitized form and is provided at a sampling rate of 8 kHz, 11 kHz or 16 kHz respectively depending on the sampling rate of the input speech from which the feature vectors have been extracted.

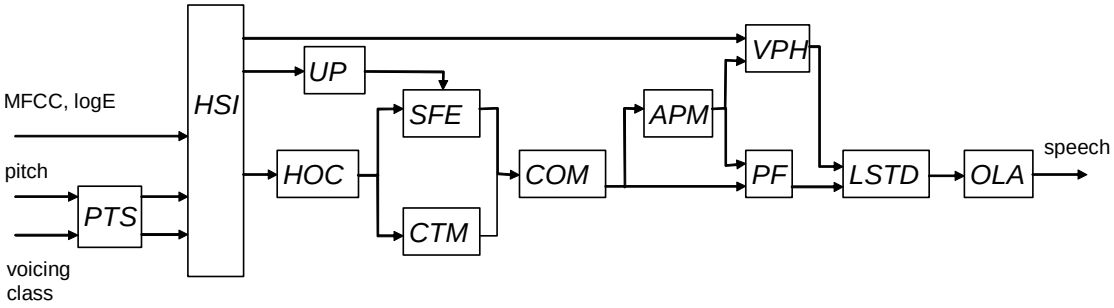
The specification also covers a pitch tracking and smoothing algorithm, which is applied to the pitch (and class) parameters before they are used for speech reconstruction.

8.2 Algorithm description

The reconstruction algorithm synthesizes one frame of speech signal from each MFCC vector and the corresponding log$E$, pitch and voicing class parameters. Frame synthesis is based on a harmonic model representation. The model parameters, viz., harmonic frequencies, magnitudes, and phases, are estimated for each frame and a complex spectrum (STFT) of the frame is computed. The complex spectrum is then transformed to time-domain representation and overlap-added with part of the speech signal already synthesized.

8.2.1 Speech reconstruction block diagram

Speech reconstruction block diagram is shown in figure 8.1.



APM	All-Pole spectral envelope Modelling
COMB	Combined magnitudes estimate calculation
CTM	Cepstra To Magnitudes transformation
HOCR	High Order Cepstra Recovery
HSI	Harmonic Structure Initialization
LSTD	Line Spectrum to Time-Domain transformation
OLA	OverLap-Add
PF	PostFiltering
PTS	Pitch Tracking and Smoothing
SFEQ	Solving Front-End eQuation
UPH	Unvoiced PHase synthesis
VPH	Voiced PHase synthesis

Figure 8.1: Speech reconstruction block diagram

8.2.2 Pitch tracking and smoothing

The input to the Pitch Tracking and Smoothing block (PTS) is a set of successive pitch period values $P[n]$, log energy values $logE[n]$ and voicing class values $VC[n]$. (Zero pitch period indicates either an unvoiced frame or non-speech frame.) The outputs are the corrected values $p_{fixed}\,[n]$ **of pitch period** and $vc_{fixed}\,[n]$ of voicing class.

Pitch processing is done in three stages. Then the voicing class value correction is performed.

The three stages of pitch processing require three working buffers to hold the pitch values of successive frames and possibly the log-energy of the frames (for the first stage only). Each stage introduces further delay (look-ahead) in the output pitch value. The buffer length *L* (an integer number of frames) is the sum of the number of look-ahead frames (the delay) *D*, the number of backward frames (the history) *H*, plus one which is the current output frame at that stage (i.e. *L = D + H + 1*). Each stage produces a new output value, which is pushed at the top (at the end) of the next stage buffer. All other values in the buffer are pushed one frame backwards, with the oldest value discarded. This configuration is described in figure 8.2.

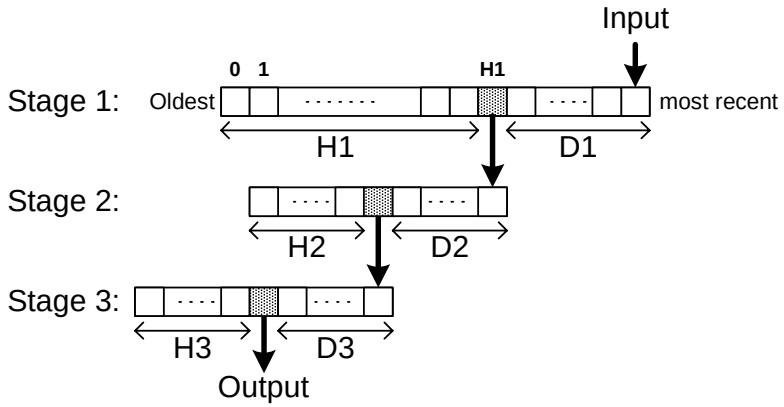


Figure 8.2: Buffers of the three-stage pitch tracking and smoothing algorithm

The total look-ahead (in frames) required for the correction of current pitch value, and therefore the delay introduced by the PTS block is: *D=D1+D2+D3*. The delay and history values used are:

First stage: *D1=8, H1=10 (therefore L1=19);*

Second stage: *D2=H2=1 (therefore L2=3);*

Third stage: *D3=H3=2 (therefore L3=5).*

And the total delay is **11 frames**.

All the three stage buffers are initialized by zero values. Each coordinate of the energy buffer used at the first stage is initialized by -50.

In the description of the three-stage pitch tracking algorithm the terms "voiced frame" and "unvoiced frame" are redefined. A frame is referred to as voiced frame if it is either of "*fully voiced*" or of "*mixed-voiced*" class. A frame is referred to as unvoiced if it is of "*unvoiced*" or "*non-speech*" class.

8.2.2.1 First stage - gross pitch error correction

Let *p[n], n=0, 1, ..., L1-1* be the pitch period values of the *first stage buffer*, such that *p[L1-1]* is the most recent value (the new input pitch), and *p[0]* is the oldest value. A pitch value of zero indicates an unvoiced frame. Similarly, there is a buffer of the same length holding the energy values.

The output pitch of the first stage has a delay of *D1* frames compared to the most recent frame in the buffer. The processed frame has *D1* frames look-ahead and *H1* backwards frames. A new pitch value P_{out} associated with the location *n=H1* in the buffer has to be calculated and pushed to the second stage pitch tracking.

If the frame is unvoiced (i.e. *p[H1]==0*) then $P_{out}=0$ as well.

If the frame is voiced, but there are unvoiced frame at both sides (i.e. *p[H1]!=0, p[H1-1]==p[H1+1]==0*), then $P_{out}=0$.

If the frame is voiced, and is a member of a voiced segment of only two frames, then the similarity between the pitch values of the two voiced frames is examined as described below. If they are *similar*, then no change is made to the pitch value, i.e. $P_{out}=p[H1]$. Otherwise, the frame is reclassified as unvoiced, $P_{out}=0$.

In the remaining cases, the output pitch value P_{out} will be assigned the value *p[H1]*, or it may be assigned an integer multiplication or integer divide of *p[H1]*. To do this, first the voiced segment in which the frame *H1* is located in is identified. This voiced segment can extend *D1* frames ahead and *H1* frames backwards at the most. It will be shorter if there are unvoiced frames in the buffer. Then, a *reference* pitch value is extracted using the information from the neighbouring frames in the voiced segment. Finally, the output pitch value of the first stage is identified.

Similarity measure.

Two (positive) pitch periods P_1 and P_2 are declared as similar if for a given *similarity factor* $\rho > 1$ the following is true:

$$\rho \times P_1 \geq P_2 \geq P_1 / \rho$$

A similarity factor of 1,28 is used to check the similarity of two pitch periods of successive frames (i.e. 10 ms apart). A factor of 1,4 is used for pitch periods that are two frames apart (20 ms).

Relevant frames identification.

The voiced segment in which the current frame (in position *H1*) is located and its pitch and energy values are copied to a temporary buffer. The pitch values of this segment are notified by *q[n], n=0,1,…N-1* and the corresponding log-energy values as *e[n], n=0,1,…,N-1*. Here *N* is the number of frames in the voiced segment. (Note that $2 < N \leq L1$.) Figure 8.3 describes the indexing of the voiced segment. "U" represents an unvoiced frame, and "V" a voiced frame. Location *K* in the voiced segment now represents the current examined frame (*p[H1]*, for which a first stage output pitch value must be calculated).

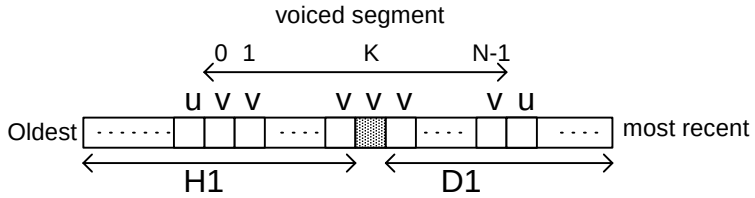


Figure 8.3: Location of a voiced segment within the first stage buffer

The purpose of the following process is to identify the set of frames that have *similar* pitch values, and their total energy is the greatest. To do that, the *N* pitch values are sorted according to ascending pitch values. The sorted pitch values are then divided into groups. A group contains one or more consecutive sorted pitch periods, such that neighbouring pitch values are *similar* (with the similarity factor 1,28) in the sense defined above. The pitch values are processed from the smallest to the largest. When the similarity is violated between the consecutive sorted pitch values, the previous group is closed and a new group is opened.

For each group, the total energy of all frames in the group is calculated. The group that has the biggest total energy is selected. All other frames that are not within the selected group are marked as deleted in the original (unsorted) voiced segment temporary buffer *q*.

Reference pitch value calculation.

One or more pitch tracks are identified in the voiced segment (represented by the buffers *q* and *e*). The tracking is done only on the frames that were not deleted by the relevant frames identification process. If frame *K* (examined frame of the stage 1) was not deleted, it will be included in one of the pitch tracks. A pitch track is defined as a set of successive undeleted voiced frames, whose neighbouring pitch values are *similar* in the above specified sense. The energy of each pitch track is the sum of the log-energy of all its frames.

After all the pitch tracks are identified, the one with the biggest energy is examined. The *reference* pitch P_{ref} is defined as the pitch value in the selected track that is closest to position *K*. If the selected pitch track includes frame *K*, it means that the reference pitch is exactly the pitch value of the examined frame (meaning it will not change at the first stage of processing).

First stage output calculation.

Let p_1 and p_2 be two positive numbers. We define the distance measure $Dist(p_1,p_2)$ in the following way:

$$Dist(p_1, p_2) = \left| \frac{p_1 - p_2}{p_1 + p_2} \right|$$

Given a reference pitch value P_{ref} and the pitch value of the current examined frame $p[H1]$, the new pitch value P_{out} is calculated as specified by the following pseudo code:

```
INTEGER SCALING
{
    if (P_ref == p[H1])
        P_out == p[H1]) ;
    elseif (P_ref > p[H1])
    {
        Q = ceil(P_ref / p[H1]) ;
        M = arg min_{m=1,...,Q} Dist(P_ref , m× p[H1]);
        P_out = M × p[H1] ;
    }
else
    {
        Q = ceil( p[H1]/ P_ref ) ;
        M = arg min_{m=1,...,Q} Dist(m× P_ref , p[H1]);
        P_out = p[H1] / M ;
    }
        if (M == 2)
    {
        if (P_ref > p[H1])
        {
            if (1,4× Dist(P_ref ,2 p[H1])> Dist(P_ref , p[H1]))
            P_out = p[H1] ;
        }
        if (P_ref < p[H1])
        {
            if (1,4× Dist(2P_ref , p[H1])> Dist(P_ref , p[H1]))
            P_out = p[H1] ;
        }
    }
}
```

8.2.2.2 Second stage - voiced/unvoiced decision and other corrections

Let *p[n], n=0,1,2 (L2=3)* be the pitch period values of the *second stage buffer*, such that *p[2]* is the most recent value (the new output of the first stage), and *p[0]* is the oldest value. An output value will be associated with the middle location *n=1* in the buffer, and will be marked P_{out}.

P_{out} will be assigned the value of *p[1]*, unless one of the following occurs:

- If all three frames are voiced, and *p[2]* is *similar* to *p[0]*, then we examine the middle value *p[1]*. If it is not *similar* (with $\rho = 1{,}28$) to the average of *p[2]* and *p[0]*, the output value P_{out} will receive this average value instead of *p[1]*.

- If *p[0]* and *p[2]* are voiced and *similar*, and if *p[1]* is unvoiced, then the output frame will be voiced with a pitch P_{out} equal to average of *p[0]* and *p[2]*. Here the similarity is evaluated using a similarity factor of $\rho = 1{,}28$ instead of 1,4, even though the pitch values to be compared are two frames apart.

- If the oldest frame in the buffer is unvoiced (*p[0]==0*) and the two other frames are voiced, or if the most recent frame is unvoiced (*p[2]==0*) and the two other frames are voiced, then the *similarity* between the two voiced frames is examined. If they are not *similar*, then the output frame will be unvoiced, i.e. $P_{out}=0$.

8.2.2.3 Third stage - smoothing

Let *p[n], n = 0, 1, ..., L3 - 1* be the pitch period values of the *third stage buffer*, such that *p[L3-1]* is the most recent value (the new output of the second stage), and *p[0]* is the oldest value. *L3* is odd. An output value will be associated with the middle location *(L3-1)/2* in the buffer, and will be marked p_{fixed}.

If there is an unvoiced frame in the middle location (i.e. *p[(L3-1)/2]==0*) then the output frame is also unvoiced and $p_{fixed}=0$. Otherwise, a filtering operation is performed by weighting a modified version of all the pitch values in the buffer as described below.

A new set of pitch values *q[n], n = 0, 1, …, L3 - 1* is derived from the current values *p[n]* in the third stage buffer, according to the following rules:

- *q[(L3-1)/2] = p[(L3-1)/2]*.

- For each n, if *p[n]==0* (unvoiced frame) then *q[n] = p[(L3-1)/2]*.

- All other pitch values are multiplied by an integer or divided by an integer, such that they become as close as possible to the value of the middle frame *p[(L3-1)/2]*. That is, *q[n] = M×p[n]* or *q[n]=p[n]/M* where M is an integer greater or equal one. The exact calculation of the new value is done as is described by the pseudo code titled INTEGER SCALING in the clause 8.2.2.1 above wherein the variables substitution should be done as: P_{ref} *by p[(L3-1)/2], p[H1] by p[n], and* P_{out} *by q[n]*.

The final output pitch is calculated in the following way:

$$p_{fixed} = \sum_{n=0}^{L3-1} q[n] \times h[n]$$

where:

$$h[0]=1/9,\ h[1]=2/9,\ h[2]=3/9,\ h[3]=2/9,\ h[4]=1/9,\ \text{(L3 = 5)}.$$

8.2.2.4 Voicing class correction

The input for the voicing class correction are three voicing class values *VC[n-1], VC[n]* and *VC[n+1]* associated with three consecutive frames, and pitch values before and after the tracking procedure associated with the middle frame *n* and marked as *P* and p_{fixed} correspondingly. The output of this processing step is a corrected voicing class value vc_{fixed} associated with the middle frame *n*. *VC[n-1]* is initialized by zero when the very first frame is processed. The processing is described by the following pseudo code:

```
{
    if (VC[n-1]=="mixed-voiced" AND VC[n]=="fully-voiced" AND VC[n+1] != "fully-voiced")
        vc_fixed = "mixed-voiced";
    else
        vc_fixed = VC[n];

    if (P == 0  AND  p_fixed != 0)
        vc_fixed = "mixed-voiced";
    elseif (P != 0  AND  pfixed == 0)
        vc_fixed = "unvoiced";
}
```

8.2.3 Harmonic structure initialization

Inputs for the harmonic structure initialization (HSI) block are the pitch value p_{fixed} and the voicing class value vc_{fixed} corresponding to the current frame being synthesized. The HSI block produces modified values of the input parameters and array(s) of harmonic-elements.

The pitch value representing a period duration in 8kHz samples is transformed to the actual sampling rate:

$$p = p_{fixed} \times f_s / 8 \tag{8.1}$$

The reconstruction algorithm treats non-speech frames and unvoiced frames in the same way. Consequently the voicing class value is modified as:

```
if (vc_fixed == "non-speech")
    vc = "unvoiced";
else
    vc = vc_fixed;
```

The modified voicing class *vc* has one of the three possible values: "*fully-voiced*", "*mixed-voiced*", and "*unvoiced*". Accordingly we refer to the frame being synthesized as fully-voiced, mixed-voiced or unvoiced.

For a fully-voiced frame an array $VH = \{H_k, k=1,\ldots,N_v\}$ of harmonics is allocated. Each harmonic $H_k = (f_k, A_k, \varphi_k)$ is represented by a normalized frequency f_k, magnitude A_k and phase φ_k values. The number of harmonics N_v is:

$$N_v = floor(p/2) \tag{8.2}$$

The normalized frequency f_k associated with k-th harmonic is set to:

$$f_k = k/p \tag{8.3}$$

For an unvoiced frame an array $UH = \{H_k, k=1,\ldots,N_u\}$ of harmonics is allocated. The number of harmonics N_u is:

$$N_u = FFTL/2 - 1 \tag{8.4}$$

The normalized frequency associated with k-th harmonic is set to:

$$f_k = k/FFTL \tag{8.5}$$

For a mixed-voiced frame both *VH* and *UH* arrays are allocated.

The *HSI* block does not set values of the harmonic magnitudes and phases. This is a subject of the further processing.

The elements of the *VH*-array will be henceforth referred to as voiced harmonics, and the elements of the *VU*-array as unvoiced harmonics.

8.2.4 Unvoiced Phase (UPH) synthesis

The input for the Unvoiced Phase synthesis (UPH) block is the UH array of unvoiced harmonics. Thus the block is entered only if the *vc*_variable value is either "unvoiced" or "mixed-voiced". The block sets phase values $\{\varphi_k, k=1,\ldots,N_u\}$ associated with the array elements (unvoiced harmonics). The phase values are obtained by a generator of pseudo random uniformly distributed numbers, and they are scaled to fit into the interval [0π, 2π]. A new vector of phase values is generated each time the UPH block is entered.

8.2.5 Harmonic magnitudes reconstruction

Harmonic magnitudes reconstruction is done in three major steps. An estimate A^E of the magnitudes vector is obtained in the *SFEQ* block. Another estimate A^I of the magnitudes vector is obtained in the *CTM* block. Then a final estimate A is calculated in the *COMB* block by combining A^E with A^I.

8.2.5.1 High order cepstra recovery

The harmonic magnitudes are estimated from the Mel-Frequency Cepstral Coefficients (MFCC) and the pitch period value (clauses 8.2.5.2 to 8.2.5.4). At the front-end, only 13 of the 23 possible MFCC's are computed (clause 4.2.11), compressed, and transmitted to the back-end. The remaining 10 values, C_{13} through C_{22}, referred to as high order cepstra here, are simply discarded, i.e. not computed. Clearly, if these missing values are available, the harmonic magnitudes can be estimated more accurately. The HOCR block attempts to at least partially recover the missing high order cepstral information for voiced frames (both mixed and fully voiced). This recovery process continues further within the Solving Front-Equation (SFEQ) block as described below in clause 8.2.5.2. For unvoiced frames, the high order cepstra are not recovered.

The recovery of high order cepstra is achieved through lookup tables (tables 8.1 to 8.3) using the pitch period as a parameter. These tables were generated by analysing a large speech database and computing the average value of (uncompressed) high order cepstra over all frames with pitch values falling in the appropriate range.

Table 8.1: High order cepstra for different pitch ranges (8 kHz sampling)

Pitch Range	C_{13} thru C_{22}	Pitch range	C_{13} thru C_{22}	Pitch range	C_{13} thru C_{22}	Pitch range	C_{13} thru C_{22}
	3,294131E+00		-2,988516E-01		1,752766E-01		-1,393523E-01
	1,879188E+00		-8,849008E-01		-1,175471E-01		-1,065410E+00
	9,031433E-01	40 <	-5,150088E-01	53 <	4,111699E-01	67 <	6,370883E-02
p	-6,946425E-01	p	-8,201993E-01	p	4,214908E-01	p	-5,321652E-01
≤ 25	-2,741839E-01	≤ 41	-9,407813E-01	≤ 54	5,142949E-01	≤ 68	3,508367E-01
	9,019766E-01		-1,197148E+00		2,668873E-01		-2,175000E-01
	1,502901E+00		-7,192041E-01		2,669893E-01		1,575631E-01
	5,457747E-01		-2,216629E-01		-2,565804E-02		-1,793344E-01
	2,176694E-01	41 <	2,389476E-01	54 <	4,410679E-02	68 <	-9,548394E-02
25 <	5,524537E-01	p	2,304727E-01	p	5,593655E-04	p	-1,626251E-01
p		≤ 42		≤ 55		≤ 69	
≤ 29	-3,517185E-01		-1,692750E-01		1,542231E-01		-6,432290E-02
	-1,103502E+00		-7,110255E-01		-1,363242E-01		-1,072603E+00
	-2,396688E-01		-3,592656E-01		5,465631E-01		2,318707E-02
	-5,023954E-01	42 <	-5,792361E-01	55 <	4,052597E-01	69 <	-5,277231E-01
29 <	-6,816051E-02	p	-9,034623E-01	p	4,404614E-01	p	2,949342E-01
p	-2,993481E-01	≤ 43	-1,027697E+00	≤ 56	2,150040E-01	≤ 70	-2,091611E-01
≤ 30	1,310237E-02		-7,302259E-01		3,143147E-01		1,345669E-01
	-1,283895E-01		-4,110435E-01		4,376205E-02		-1,956731E-01
	-2,207362E-02		-1,232714E-01		1,059225E-01		-8,480399E-02
	-4,813484E-02	43 <	-4,008306E-02	56 <	-2,500212E-03	70 <	-1,673282E-01
30 <		p		p		p	
p	-1,393348E+00	≤ 44	-1,372496E-01	≤ 58	2,732437E-01	≤ 71	-5,622257E-02
≤ 31	-9,585445E-01		-6,080195E-01		-3,639484E-01		-1,076187E+00
	7,983048E-01		-2,807112E-01		6,189037E-01		-6,939828E-02
	1,146215E+00		-4,264785E-01		2,314612E-01		-4,716512E-01
	8,490435E-01	44 <	-6,480206E-01	58 <	5,251961E-01	71 <	1,736834E-01
31 <	-1,313370E-01	p	-8,858252E-01	p	2,298868E-01	p	-1,633613E-01
p	-1,348895E+00	≤ 45	-6,590320E-01	≤ 59	3,155471E-01	≤ 72	1,204333E-01
≤ 32	-1,489402E+00		-5,895118E-01		6,287218E-02		-1,988844E-01
	-3,404197E-01		-3,504879E-01		1,176859E-01		-9,681459E-02
	3,550971E-01		-1,844929E-01		1,915457E-02		-1,600474E-01
	-1,806770E+00	45 <	2,773138E-02	59 <	2,744928E-01	72 <	-1,387690E-01
32 <	-1,593115E+00	p	-5,146922E-01	p	-4,139554E-01	p	-1,106376E+00
p	-2,750653E-02	≤ 46	5,568316E-03	≤ 60	3,776372E-01	≤ 73	-7,631757E-02
≤ 33	6,877274E-01		-3,666849E-01		1,759360E-01		-4,575221E-01
	8,805254E-01		-4,899669E-01		5,221393E-01		6,408327E-02
	1,137417E-01		-7,897312E-01		1,249147E-01		-1,516200E-01
	-8,289580E-01	46 <	-6,561837E-01	60 <	3,310523E-01	73 <	1,511662E-01
33 <	-1,188777E+00	p	-6,008886E-01	p	4,418805E-02	p	-1,819421E-01
p	-3,945412E-01	≤ 47	-3,987320E-01	≤ 61	7,021046E-02	≤ 74	-6,963367E-02
≤ 34	1,272422E-01		-1,921384E-01		5,348071E-02		-1,752035E-01
	-2,109219E+00		9,633469E-02		1,091691E-02		-3,357490E-01
	-2,148034E+00	47 <	-4,281098E-01	61 <	-4,028163E-01	74 <	-1,040818E+00
34 <	-6,176175E-01	p	1,519668E-01	p	4,389749E-01	p	-7,767339E-02
p	3,498008E-01		-1,898231E-01		5,693171E-02		-4,978964E-01

Pitch Range	C_{13} thru C_{22}	Pitch range	C_{13} thru C_{22}	Pitch range	C_{13} thru C_{22}	Pitch range	C_{13} thru C_{22}
≤ 35	7,252700E-01	≤ 48	-4,036781E-01	≤ 62	4,589326E-01	≤ 75	7,351060E-02
	4,283130E-01		-6,767984E-01		1,050263E-01		-2,345509E-01
	-1,095525E-01		-6,498718E-01		2,530933E-01		1,979697E-01
	-6,075157E-01		-6,409187E-01		3,224460E-02		-2,042752E-01
35 <	-5,468144E-01	48 <	-4,282145E-01	62 <	7,309323E-02	75 <	-3,967336E-02
p	-3,470269E-01	p	-2,063774E-01	p	6,794002E-03	p	-1,729749E-01
≤ 36		≤ 49		≤ 63		≤ 76	
	-2,328319E+00		5,128368E-02		1,553257E-01		-3,363196E-01
	-2,647131E+00		-3,322522E-01		-5,752975E-01		-1,074411E+00
	-1,144696E+00		2,163569E-01		4,054688E-01		-2,692610E-02
	6,951346E-03		-5,868236E-02		1,875337E-02		-5,193648E-01
36 <	5,625280E-01	49 <	-1,891985E-01	63 <	5,092713E-01	76 <	7,648631E-02
p	5,844895E-01	p	-5,643914E-01	p	6,860561E-02	p	-2,812090E-01
≤ 37	6,877614E-01	≤ 50	-5,151935E-01	≤ 64	2,508796E-01	≤ 77	1,832394E-01
	-6,197921E-02		-6,299446E-01		-3,798583E-02		-1,808847E-01
	-6,763814E-01		-4,192819E-01		8,072541E-02		-2,877708E-02
	-7,744182E-01		-2,077382E-01		1,236025E-02		-1,580936E-01
37 <		50 <		64 <		77 <	
p	-2,285553E+00	p	1,461182E-01	p	-2,376247E-02	p	-2,988284E-01
≤ 38	-2,963302E+00	≤ 51	-2,911482E-01	≤ 65	-5,097925E-01	≤ 78	-1,103823E+00
	-1,709311E+00		3,839266E-01		3,516957E-01		-7,370897E-02
	-3,844131E-01		2,960237E-02		9,448875E-02		-5,530614E-01
	3,183937E-01		-5,351829E-02		4,472362E-01		1,833754E-01
38 <	7,880205E-01	51 <	-3,594112E-01	65 <	1,127977E-02	78 <	-3,555229E-01
p	1,181769E+00	p	-4,609720E-01	p	2,190847E-01	p	1,393937E-01
≤ 39	5,101148E-01	≤ 52	-6,115775E-01	≤ 66	2,546498E-02	≤ 79	-1,843326E-01
	-5,748582E-01		-4,624372E-01		4,532406E-02		-1,045751E-02
	-8,982629E-01		-2,581518E-01		2,105376E-02		-1,086657E-01
	-2,209689E+00		2,588957E-01		-3,531601E-02		-3,463852E-01
39 <	-2,738691E+00	52 <	-2,987848E-01	66 <	-5,711497E-01	79 <	-1,078285E+00
p	-1,912894E+00	p	4,147992E-01	p	2,974023E-01	p	-5,001788E-02
≤ 40	-8,599744E-01	≤ 53	1,316073E-01	≤ 67	-2,028426E-03	≤ 80	-5,862272E-01
	4,213742E-02		1,505384E-01		4,524812E-01		1,787949E-01
	7,944544E-01		-2,270889E-01		-6,431431E-02		-3,795385E-01
	1,300180E+00		-2,973209E-01		1,546494E-01		9,429462E-02
	6,619403E-01		-5,367760E-01		-5,980445E-02		-1,805439E-01
	-2,124371E-01		-4,535146E-01		4,092953E-02	80 <	-2,699836E-02
	-6,700241E-01		-3,024792E-01		2,486595E-03	p	-6,708920E-02
	-1,661505E+00		2,297968E-01		2,147347E-02		-1,634159E-01
	-2,372858E+00		-2,975205E-01		-6,174898E-01		-1,253244E+00
	-1,909757E+00		4,372300E-01		1,995171E-01		-2,401476E-02
	-1,278248E+00		2,196309E-01		-1,478089E-01		-7,135424E-01
	-2,025693E-01		2,414933E-01		4,182756E-01		1,299647E-01
	5,020906E-01		-1,025894E-01		-6,741808E-02		-3,569654E-01
	1,144470E+00		-1,543293E-01		1,509666E-01		3,508006E-02
	6,584050E-01		-4,685154E-01		-9,877002E-02		-1,579050E-01
	1,422871E-01		-3,766788E-01		5,548984E-02		-3,878338E-02
	-1,906929E-01		-2,629635E-01		-1,516375E-02		-6,297522E-02
	-1,323904E+00		2,877707E-01		-4,881799E-02		-5,496578E-02
	-1,767607E+00		-2,252190E-01		-7,272697E-01		-1,314187E+00
	-1,779609E+00		4,797849E-01		2,093155E-01		-2,389390E-02
	-1,379924E+00		2,568238E-01		-1,675496E-01		-7,930011E-01
	-6,683853E-01		3,591473E-01		3,912781E-01		6,372795E-02
	1,410876E-01		-6,040132E-03		-9,287293E-02		-4,334946E-01
	6,331957E-01		-1,599351E-02		1,142080E-01		5,706436E-02
	5,338485E-01		-3,189542E-01		-1,058651E-01		-1,314620E-01
	4,527746E-01		-3,349573E-01		3,111744E-03		-3,085178E-02
	2,962149E-01		-2,167672E-01		-2,473840E-02		-4,978850E-02
	-8,933408E-01		2,098745E-01		-1,233369E-01		-4,740081E-02
	-1,459709E+00		-2,310972E-01		-8,680894E-01		-1,408440E+00
	-1,310935E+00		5,116026E-01		2,350561E-01		1,902080E-02
	-1,384499E+00		2,493115E-01		-3,523748E-01		-8,696354E-01

Pitch Range	C_{13} thru C_{22}	Pitch range	C_{13} thru C_{22}	Pitch range	C_{13} thru C_{22}	Pitch range	C_{13} thru C_{22}
	-9,453042E-01		4,376403E-01		3,619746E-01		6,922893E-02
	-4,160211E-01		1,159550E-01		-9,724603E-02		-5,147804E-01
	1,477768E-01		1,265852E-01		1,535049E-01		1,066057E-02
	2,974642E-01		-2,158664E-01		-1,435163E-01		-1,066578E-01
	6,838222E-01		-2,126734E-01		-4,990274E-02		-2,634774E-02
	6,062507E-01		-1,743655E-01		-2,600003E-02		-2,722837E-02
	-7,088845E-01		2,331659E-01		-1,284105E-01		-6,620023E-02
	-1,230544E+00		-1,213713E-01		-9,707058E-01		-1,405944E+00
	-9,357001E-01		5,016556E-01		2,352113E-01		-8,409542E-03
	-1,034249E+00		3,452590E-01		-3,668477E-01		-8,138001E-01
	-1,057516E+00		4,266254E-01		3,522376E-01		6,096114E-02
	-8,781891E-01		1,766788E-01		-1,655475E-01		-5,707809E-01
	-3,092404E-01		2,166634E-01		1,216727E-01		-5,838640E-02
	1,260653E-01		-1,406473E-01		-1,268461E-01		-1,276284E-01
	7,202828E-01		-9,039140E-02		-1,038426E-01		-9,554439E-03
	6,493044E-01		-1,136983E-01		-5,855845E-02		-3,698357E-03
	-5,750586E-01		1,956274E-01		-1,355420E-01		-1,253647E-01
	-1,024682E+00		-6,157059E-02		-1,138031E+00		-1,221195E+00
	-8,594348E-01		5,553829E-01		1,837333E-01		-4,560992E-02
	-8,730156E-01		3,631779E-01		-4,721983E-01		-6,385362E-01
	-1,012759E+00		4,846472E-01		3,377192E-01		1,036283E-01
	-1,139837E+00		1,054959E-01		-1,789617E-01		-4,404687E-01
	-6,131858E-01		2,815528E-01		1,788783E-01		-8,734322E-02
	-3,085556E-02		-2,832542E-02		-1,532787E-01		-1,534337E-01
	5,150890E-01		-1,050620E-02		-1,053026E-01		-5,258718E-02
	4,728776E-01		-3,441706E-02		-1,182124E-01		-3,601738E-03
							-3,517185E-01
							-1,103502E+00
							-2,396688E-01
							-5,023954E-01
							-6,816051E-02
							-2,993481E-01
							1,310237E-02
							-1,283895E-01
							-2,207362E-02
							-4,813484E-02

Table 8.2: High order cepstra for different pitch ranges (11 kHz sampling)

Pitch range	C_{13} thru C_{22}	Pitch range	C_{13} thru C_{22}	Pitch range	C_{13} thru C_{22}	Pitch range	C_{13} thru C_{22}
	5,626050E-02		6,594623E-01		-5,985192E-01		-8,699980E-01
	4,010505E-01		-4,052819E-01		-5,292195E-01		-5,130804E-01
	9,829021E-01	40 <	-5,949365E-01	53 <	-6,240759E-01	67 <	-5,766745E-01
p	8,026245E-01	p	-3,599615E-02	p	8,768867E-03	p	-4,153023E-01
≤ 25	4,193927E-03	≤ 41	-2,652443E-01	≤ 54	5,093549E-02	≤ 68	1,090937E-02
	-3,766599E-01		-6,140299E-02		1,432854E-01		-9,835050E-02
	-7,377624E-01		-2,546754E-01		3,483640E-01		5,715146E-02
	-6,102725E-01		-6,026013E-01		-1,074831E-01		-4,872324E-02
	2,690858E-02	41 <	-4,842791E-01	54 <	7,235842E-02	68 <	-7,950265E-02
25 <	3,398359E-01	p	-3,309598E-01	p	-2,788544E-02	p	-5,391960E-04
p		≤ 42		≤ 55		≤ 69	
≤ 29	-5,403826E-01		6,159833E-01		-5,348074E-01		-7,577375E-01
	-7,297218E-01		-2,166754E-01		-6,087937E-01		-5,989230E-01
	-7,981775E-01		-6,798998E-01		-6,908400E-01		-4,603150E-01
	-4,187649E-01	42 <	1,045943E-01	55 <	-8,651746E-02	69 <	-4,425568E-01
29 <	-3,296735E-01	p	-1,157796E-01	p	-1,828480E-02	p	-4,194269E-03
p	-6,503036E-02	≤ 43	1,127240E-01	≤ 56	1,647329E-01	≤ 70	-1,147042E-01
≤ 30	-3,869137E-02		-1,591593E-01		3,555764E-01		2,104535E-02
	-1,486998E-01		-4,188999E-01		-1,830124E-01		4,157012E-03
	-9,623731E-02		-4,568502E-01		9,306553E-02		-1,119670E-01
	-6,828123E-02	43 <	-2,635819E-01	56 <	-5,796085E-02	70 <	8,145819E-04
30 <		p		p		p	
p	-2,120098E+00	≤ 44	5,604003E-01	≤ 58	-4,562387E-01	≤ 71	-6,471772E-01
≤ 31	-2,603270E+00		-7,031132E-02		-7,160269E-01		-6,759631E-01
	-1,961386E+00		-4,957428E-01		-7,286356E-01		-4,054322E-01
	-6,437509E-01		3,095440E-02		-1,586156E-01		-3,794170E-01
	-1,297146E-01	44 <	-1,472567E-01	58 <	-2,752461E-02	71 <	-1,011809E-01
31 <	9,896320E-01	p	2,583736E-01	p	8,430939E-02	p	-9,334780E-02
p	7,504307E-01	≤ 45	-5,709013E-02	≤ 59	2,411337E-01	≤ 72	3,991572E-02
≤ 32	2,412817E-01		-2,277715E-01		-1,778228E-01		-1,564501E-02
	-3,295308E-01		-2,509870E-01		4,920438E-02		-9,772810E-02
	-5,005424E-01		-2,368541E-01		-3,671738E-02		-2,459401E-02
	-2,325229E+00	45 <	3,903897E-01	59 <	-4,903277E-01	72 <	-5,706857E-01
32 <	-2,704816E+00	p	-1,453898E-01	p	-7,073108E-01	p	-7,904795E-01
p	-2,361322E+00	≤ 46	-2,537936E-01	≤ 60	-6,139755E-01	≤ 73	-3,469868E-01
≤ 33	-8,539604E-01		1,069747E-01		-3,190579E-01		-3,812572E-01
	-3,659675E-01		-1,764263E-01		-1,059218E-01		-1,517786E-01
	9,933884E-01		3,500780E-01		9,667006E-03		-8,318068E-02
	8,260861E-01	46 <	5,604815E-02	60 <	1,972758E-01	73 <	3,613629E-02
33 <	4,746376E-01	p	-1,798330E-01	p	-1,227097E-01	p	-7,302897E-02
p	2,391564E-02	≤ 47	-2,168454E-01	≤ 61	3,240498E-02	≤ 74	-3,424092E-02
≤ 34	-2,725177E-01		-2,169755E-01		-4,227853E-02		-4,346391E-03
	-1,917424E+00		2,083062E-01		-6,995904E-01		-7,399834E-01
	-2,405833E+00	47 <	-2,430097E-01	61 <	-6,085395E-01	74 <	-7,475148E-01
34 <	-2,719004E+00	p	-2,983543E-01	p	-6,532067E-01	p	-3,700922E-01
p	-1,101024E+00	≤ 48	2,918483E-01	≤ 62	-2,718156E-01	≤ 75	-4,386602E-01
≤ 35	-8,479526E-01		4,478910E-02		1,181321E-02		-4,668776E-02
	7,480610E-01		3,493175E-01		-4,461569E-02		-1,611996E-01
	6,160174E-01		1,187371E-01		2,147919E-01		8,287523E-02
	5,446568E-01	48 <	-2,091469E-01	62 <	-1,297862E-01	75 <	-1,182228E-01
35 <	3,330878E-01	p	-2,651710E-01	p	8,102247E-03	p	1,637517E-02
p	5,541263E-02	≤ 49	-1,916646E-01	≤ 63	-4,309627E-02	≤ 76	-1,351070E-02
≤ 36	-1,243290E+00		7,230666E-02		-8,721040E-01		-9,118970E-01
	-1,502970E+00		-1,957404E-01		-5,641596E-01		-6,244987E-01
	-2,559822E+00		-2,398411E-01		-6,915667E-01		-4,572108E-01
	-1,132253E+00	49 <	2,689252E-01	63 <	-3,315040E-01	76 <	-4,725367E-01
	-1,416679E+00	p	1,815998E-01	p	-5,745752E-02	p	9,458130E-02
36 <	6,267955E-02	≤ 50	4,484467E-01	≤ 64	-9,005700E-02	≤ 77	-2,499902E-01
p	1,259953E-01		1,459312E-01		2,321653E-01		1,249284E-01
≤ 37	3,109652E-01		-1,775228E-01		-1,118844E-01		-1,241206E-01
	7,334375E-01		-2,158363E-01		-2,671431E-02		1,874613E-03

Pitch range	C_{13} thru C_{22}	Pitch range	C_{13} thru C_{22}	Pitch range	C_{13} thru C_{22}	Pitch range	C_{13} thru C_{22}
	3,803246E-01	50 <	-1,916764E-01	64 <	-3,152244E-02	77 <	-1,316657E-02
37 <		p		p		p	
p	-8,454075E-01	≤ 51	1,178002E-02	≤ 65	-7,592263E-01	≤ 78	-1,008204E+00
≤ 38	-1,095878E+00		-3,857245E-01		-6,956729E-01		-5,179963E-01
	-2,061551E+00		-1,687225E-01		-8,429078E-01		-4,310535E-01
	-1,216612E+00		2,721840E-01		-2,097507E-01		-4,581129E-01
	-1,412864E+00	51 <	2,954684E-01	65 <	-7,842355E-02	78 <	4,912210E-02
38 <	-4,129256E-01	p	5,211892E-01	p	-9,745515E-02	p	-1,774102E-01
p	-1,334834E-01	≤ 52	1,927191E-01	≤ 66	2,900715E-01	≤ 79	1,246857E-01
≤ 39	1,531512E-01		-1,046720E-01		-1,180824E-01		-1,000912E-01
	6,831274E-01		-9,969496E-02		-2,032569E-02		-6,473759E-02
	4,964894E-01		-1,734926E-01		-4,731339E-02		-3,739955E-02
		52 <		66 <		79 <	
39 <	-3,748133E-01	p	-1,515269E-01	p	-7,840617E-01	p	-9,691091E-01
p	-1,209845E+00	≤ 53	-4,677946E-01	≤ 67	-6,243310E-01	≤ 80	-2,597100E-01
≤ 40	-1,454594E+00		-2,049010E-01		-7,380188E-01		-6,116344E-01
	-1,211848E+00		2,981386E-01		-2,557504E-01		-3,569628E-01
	-1,344769E+00		3,095689E-01		-1,256430E-01		1,191660E-02
	-6,566734E-01		5,104501E-01		-4,517741E-03		-1,772077E-01
	-1,565127E-01		3,203006E-01		1,595486E-01	80 <	1,139856E-01
	-3,826825E-02		-9,516710E-02		-4,156975E-02	p	-8,539633E-02
	3,767977E-01		-5,627316E-03		-7,999590E-02		-1,109010E-01
	2,408732E-01		-1,145671E-01		-4,396504E-02		-3,891525E-02
	-1,808773E-01		-1,436440E-01		-9,005838E-01		-9,604611E-01
	-8,466350E-01		-5,484876E-01		-5,855779E-01		-2,683079E-01
	-1,125851E+00		-2,247336E-01		-7,674657E-01		-5,909898E-01
	-1,139614E+00		2,702981E-01		-2,292142E-01		-2,987153E-01
	-1,236157E+00		2,612287E-01		-6,842449E-02		-6,892599E-02
	-9,769015E-01		4,342051E-01		-6,982788E-02		-7,098990E-02
	-3,295964E-01		3,323809E-01		2,243374E-01		7,328811E-03
	-2,809878E-01		-9,497844E-03		-7,543965E-02		-6,670985E-02
	9,442757E-02		4,865607E-02		-7,031597E-02		-1,568488E-01
	4,975950E-02		-5,490477E-02		-5,433935E-02		-5,570233E-02
	1,510383E-02		-1,022197E-01		-1,035621E+00		-8,962249E-01
	-7,576004E-01		-4,216861E-01		-4,664645E-01		-2,966810E-01
	-8,305255E-01		-3,195037E-01		-7,430429E-01		-5,353746E-01
	-7,981105E-01		1,615909E-01		-2,328916E-01		-2,893039E-01
	-1,161772E+00		2,488543E-01		-7,303254E-02		-1,300036E-01
	-1,005999E+00		3,378116E-01		-5,782277E-02		-4,711595E-02
	-5,391761E-01		3,185624E-01		1,521992E-01		-1,376169E-01
	-5,395351E-01		3,267338E-03		-8,366722E-02		-2,657904E-02
	-1,266667E-01		9,593730E-02		-1,001086E-01		-2,249794E-01
	-1,149153E-01		-4,221760E-02		-5,336955E-02		-4,320256E-02
	2,699264E-01		-2,112814E-01		-1,042572E+00		-9,208679E-01
	-6,047796E-01		-3,465334E-01		-5,871755E-01		-4,190644E-01
	-7,770238E-01		-3,546633E-01		-7,255636E-01		-4,713449E-01
	-4,898928E-01		1,054676E-01		-2,383813E-01		-2,947722E-01
	-8,497359E-01		2,405440E-01		-5,371830E-02		-2,108513E-01
	-7,969154E-01		3,210793E-01		-7,305724E-02		-2,939518E-02
	-6,299004E-01		2,927946E-01		1,357030E-01		-2,336141E-01
	-9,179399E-01		3,703629E-03		-4,784035E-02		1,241045E-02
	-3,081430E-01		9,307355E-02		-9,422132E-02		-2,780380E-01
	-2,666699E-01		-2,213902E-02		-3,772314E-02		-4,469913E-02
	2,106783E-01		-2,539558E-01		-1,090505E+00		-8,679105E-01
	-3,506425E-01		-4,939324E-01		-5,729430E-01		-5,008469E-01
	-7,600987E-01		-3,880466E-01		-6,577579E-01		-4,234283E-01
	-3,850166E-01		-6,713597E-03		-3,419261E-01		-3,798813E-01
	-4,434027E-01		1,464662E-01		-4,422111E-03		-1,639598E-01
	-5,009282E-01		2,390897E-01		-8,696646E-02		-4,563949E-02
	-5,068779E-01		3,295784E-01		1,223338E-01		-2,887070E-01
	-9,154452E-01		-4,054515E-02		-5,105527E-02		1,419084E-02
	-5,526160E-01		8,528621E-02		-7,064462E-02		-3,245438E-01

Pitch range	C_{13} thru C_{22}	Pitch range	C_{13} thru C_{22}	Pitch range	C_{13} thru C_{22}	Pitch range	C_{13} thru C_{22}
	-3,271186E-01		-8,548085E-03		-3,095784E-02		-3,300466E-02
	3,422788E-01		-3,828481E-01		-9,286439E-01		-8,131769E-01
	-4,262303E-01		-5,052647E-01		-5,678661E-01		-6,108836E-01
	-5,822095E-01		-4,231588E-01		-6,342901E-01		-4,705512E-01
	-2,081946E-01		-6,736862E-02		-3,782880E-01		-4,122885E-01
	-3,331535E-01		1,247756E-01		3,050694E-02		-1,257324E-01
	-1,488218E-01		2,423477E-01		-8,772710E-02		-7,427499E-02
	-4,198242E-01		3,031318E-01		7,113196E-02		-2,098987E-01
	-8,042739E-01		-5,081066E-02		-5,737579E-02		1,747474E-02
	-5,660493E-01		4,424514E-02		-7,125308E-02		-3,298417E-01
	-4,048214E-01		-1,743830E-02		-5,608497E-03		-7,603588E-02
							-5,403826E-01
							-7,297218E-01
							-7,981775E-01
							-4,187649E-01
							-3,296735E-01
							-6,503036E-02
							-3,869137E-02
							-1,486998E-01
							-9,623731E-02
							-6,828123E-02

Table 8.3: High order cepstra for different pitch ranges (16 kHz sampling)

Pitch range	C_{13} thru C_{22}	Pitch range	C_{13} thru C_{22}	Pitch range	C_{13} thru C_{22}	Pitch range	C_{13} thru C_{22}
	-2,026232E+00		5,479995E-01		8,254467E-03		-1,347010E-01
	-2,110936E+00		-1,023217E-01		-6,429565E-01		5,332863E-02
	-1,063945E+00	40 <	6,846186E-01	53 <	-4,247889E-01	67 <	-5,686797E-01
p	-5,426380E-01	p	3,619581E-01	p	-5,587783E-01	p	-5,787178E-01
≤ 25	4,834915E-01	≤ 41	3,339010E-02	≤ 54	-2,917541E-01	≤ 68	1,631299E-02
	8,876385E-01		2,488271E-01		-2,871714E-01		-3,300158E-01
	5,840230E-01		3,752856E-01		7,927610E-02		-7,623082E-02
	-3,167802E-01		7,243636E-02		7,821897E-02		6,026243E-02
	-6,446556E-01	41 <	2,611414E-01	54 <	-7,620810E-02	68 <	-1,633065E-01
25 <	-7,251640E-01	p	1,127095E-01	p	1,162470E-01	p	-6,150040E-02
p		≤ 42		≤ 55		≤ 69	
≤ 29	2,621901E-02		3,741992E-01		1,779742E-01		-1,234223E-01
	-2,024138E-01		-1,758603E-01		-7,869193E-01		2,566001E-02
	-3,691150E-01		5,266045E-01		-5,577497E-01		-5,028053E-01
	-4,738192E-01	42 <	5,161305E-01	55 <	-5,632171E-01	69 <	-6,257676E-01
29 <	-3,296576E-01	p	1,012750E-01	p	-3,093369E-01	p	-1,896448E-02
p	-2,743398E-01	≤ 43	3,495263E-01	≤ 56	-2,883354E-01	≤ 70	-2,312503E-01
≤ 30	-1,026351E-01		5,645265E-01		7,307628E-02		-9,265310E-02
	-1,362258E-01		2,029514E-01		1,747146E-03		7,122856E-02
	-7,479050E-02		4,227505E-01		-1,166880E-01		-1,521138E-01
	-1,117859E-01	43 <	6,583293E-03	56 <	1,768415E-01	70 <	-8,906260E-02
30 <		p		p		p	
p	-1,333018E+00	≤ 44	3,933974E-01	≤ 58	2,812443E-02	≤ 71	-1,331304E-01
≤ 31	-1,823165E+00		-3,173743E-01		-5,131770E-01		-5,140646E-02
	-1,512190E+00		3,916825E-01		-2,936540E-01		-5,337533E-01
	-1,297547E+00		5,527308E-01		-6,485048E-01		-6,729785E-01
	-1,457952E+00	44 <	2,106767E-01	58 <	-3,752124E-01	71 <	-6,045981E-02
31 <	-4,133839E-01	p	4,178565E-01	p	-2,621343E-01	p	-1,587423E-01
p	-8,237351E-02	≤ 45	5,196628E-01	≤ 59	-1,606354E-02	≤ 72	-3,304247E-02
≤ 32	3,371675E-01		2,104156E-01		5,167165E-02		4,030590E-02
	7,079317E-01		3,976243E-01		-1,251145E-01		-1,569136E-01
	3,336488E-01		-3,269132E-02		4,226151E-02		-6,401396E-02
	-8,177810E-01	45 <	4,069968E-01	59 <	6,459653E-02	72 <	-2,151330E-01
32 <	-1,322071E+00	p	-3,091523E-01	p	-3,181205E-01	p	-5,420145E-02
p	-1,786135E+00	≤ 46	2,340520E-01	≤ 60	-2,972990E-01	≤ 73	-4,937927E-01
≤ 33	-1,186759E+00		3,626993E-01		-7,132380E-01		-7,014854E-01
	-1,955396E+00		1,834435E-01		-2,899593E-01		-1,462503E-01
	-5,283016E-01		5,810700E-01		-2,613601E-01		-7,300983E-02
	-5,387481E-01	46 <	4,617110E-01	60 <	-2,048229E-02	73 <	-6,947362E-02
33 <	-7,300678E-02	p	2,182536E-01	p	1,386128E-02	p	1,619861E-02
p	2,830981E-01	≤ 47	3,645189E-01	≤ 61	-1,571951E-01	≤ 74	-1,612242E-01
≤ 34	4,506301E-02		2,682160E-02		3,414188E-02		-7,362535E-02
	-2,879547E-01		5,286090E-01		-2,938032E-02		-1,850086E-01
	-6,849482E-01	47 <	-3,295684E-01	61 <	-3,517489E-01	74 <	-1,228883E-01
34 <	-1,839059E+00	p	6,119184E-02	p	-3,225757E-01	p	-5,764287E-01
p	-8,057479E-01	≤ 48	1,075905E-01	≤ 62	-6,503325E-01	≤ 75	-7,236682E-01
≤ 35	-2,405975E+00		6,069104E-02		-1,280458E-01		-1,141678E-01
	-8,800773E-01		5,106977E-01		-2,856279E-01		-1,804202E-01
	-1,055467E+00		5,016756E-01		-8,394262E-02		-5,176742E-02
	-6,326932E-01	48 <	3,231877E-01	62 <	1,103418E-01	75 <	1,043618E-01
35 <	-3,407662E-01	p	3,332683E-01	p	-1,845763E-01	p	-2,352337E-01
p	-2,728538E-01	≤ 49	1,133187E-01	≤ 63	2,918438E-02	≤ 76	-3,331541E-02
≤ 36	-3,031390E-01		2,951866E-01		1,029596E-01		-1,414134E-01
	-3,394979E-01		-3,018426E-01		-1,769495E-01		-2,516573E-02
	-1,559689E+00		1,664823E-02		-4,088103E-01		-6,702498E-01
	-4,456815E-01	49 <	4,222289E-03	63 <	-7,089485E-01	76 <	-7,110191E-01
36 <	-2,036168E+00	p	-1,608906E-02	p	-8,953595E-02	p	-5,645533E-02
p	-9,838692E-01	≤ 50	4,029250E-01	≤ 64	-3,557198E-01	≤ 77	-3,436754E-01
≤ 37	-1,364973E+00		4,914628E-01		-2,068182E-02		-3,591015E-02
	-1,186102E+00		4,173051E-01		7,288132E-03		1,738667E-01
	-7,846482E-01		4,092605E-01		-1,878297E-01		-2,775487E-01

Pitch range	C_{13} thru C_{22}	Pitch range	C_{13} thru C_{22}	Pitch range	C_{13} thru C_{22}	Pitch range	C_{13} thru C_{22}
	-6,635148E-01	50 <	1,185967E-01	64 <	6,942376E-02	77 <	2,315980E-02
37 <		p		p		p	
p	-2,583467E-01	≤ 51	4,197218E-01	≤ 65	5,695233E-02	≤ 78	-6,419491E-02
≤ 38	-6,506612E-01		-4,477604E-01		-2,196874E-01		1,554566E-01
	-1,132547E+00		-4,836996E-03		-2,921967E-01		-6,197453E-01
	-3,585654E-01		-1,635454E-01		-6,754194E-01		-6,849180E-01
	-1,428531E+00	51 <	-1,377630E-01	65 <	-2,667778E-01	78 <	2,593279E-02
38 <	-9,583033E-01	p	3,034003E-01	p	-3,072328E-01	p	-3,850149E-01
p	-1,258072E+00	≤ 52	3,827793E-01	≤ 66	-2,800013E-03	≤ 79	-3,179316E-02
≤ 39	-1,341744E+00		5,153344E-01		-1,629100E-02		1,340202E-01
	-9,579476E-01		4,135734E-01		-2,071333E-01		-1,950800E-01
	-5,344057E-01		1,241358E-01		5,440867E-02		3,317590E-02
		52 <		66 <		79 <	
39 <	1,615421E-01	p	3,678487E-01	p	2,049767E-01	p	1,703393E-03
p	-1,043571E+00	≤ 53	-4,631278E-01	≤ 67	-1,025310E-01	≤ 80	1,975255E-01
≤ 40	-6,740458E-01		-9,326042E-02		-4,571202E-01		-5,575346E-01
	-6,546493E-01		-3,737336E-01		-6,315088E-01		-6,005854E-01
	-8,875172E-01		-1,472778E-01		-2,798402E-01		9,024668E-02
	-8,743641E-01		1,595358E-01		-3,037064E-01		-4,556844E-01
	-1,025171E+00		3,695181E-01		-1,274799E-02	80 <	4,772030E-02
	-1,196147E+00		4,490506E-01		1,483960E-02	p	5,254362E-02
	-8,529125E-01		3,800947E-01		-1,397844E-01		-1,584285E-01
	-3,839518E-01		1,774867E-01		3,804585E-03		-2,753272E-03
	4,734923E-01		3,615054E-01		2,284569E-01		-8,471422E-02
	-8,380715E-01		-4,550473E-01		-1,773758E-01		2,237505E-01
	-4,175060E-01		-1,407069E-01		-5,961479E-01		-4,772195E-01
	-6,309476E-01		-3,944382E-01		-6,358854E-01		-5,604665E-01
	-6,621496E-01		-1,805678E-01		-2,213772E-01		1,566324E-01
	-6,967811E-01		9,924901E-02		-3,470830E-01		-4,455221E-01
	-9,083475E-01		2,756066E-01		1,282147E-02		1,073800E-01
	-8,838883E-01		3,483957E-01		7,015398E-02		-3,147756E-02
	-8,056732E-01		2,401312E-01		-2,457183E-01		-4,447187E-02
	-3,879474E-01		1,841012E-01		2,583304E-02		-8,178144E-02
	3,484700E-01		2,330553E-01		2,688840E-01		-1,074359E-01
	-5,704010E-01		-3,904562E-01		-6,941569E-02		3,364727E-01
	-1,571234E-02		-2,328840E-01		-5,795702E-01		-3,695723E-01
	-4,434779E-01		-3,683225E-01		-6,240800E-01		-6,287698E-01
	-6,717660E-01		-3,040399E-01		-2,282836E-01		2,465456E-01
	-4,207197E-01		5,676054E-02		-3,560088E-01		-4,421380E-01
	-4,787949E-01		1,960860E-01		4,549953E-02		1,804006E-01
	-7,055746E-01		3,220345E-01		1,014333E-02		-1,223475E-01
	-6,342757E-01		1,480135E-01		-2,220406E-01		9,988480E-02
	-2,165355E-01		1,460853E-01		-2,071246E-03		-1,939847E-01
	3,611920E-01		2,239470E-01		9,385106E-02		-6,456530E-02
	-4,334970E-01		-6,513079E-01		-1,432743E-02		5,218610E-01
	2,397788E-01		-2,281708E-01		-6,293171E-01		-2,695373E-01
	-1,130040E-01		-3,713762E-01		-6,208313E-01		-8,004482E-01
	-5,057448E-01		-1,974930E-01		-1,935377E-01		3,161954E-01
	-3,094423E-01		-6,694008E-02		-3,572474E-01		-4,450321E-01
	-8,990173E-02		1,560454E-01		7,287241E-03		2,063266E-01
	-2,442444E-01		2,206244E-01		7,610204E-02		-2,401241E-01
	-2,962797E-01		6,276750E-02		-1,676478E-01		2,078162E-01
	-6,834994E-02		1,126685E-01		-3,569930E-02		-2,956866E-01
	4,358751E-01		2,433587E-01		1,029177E-01		-3,603708E-03
	-2,529008E-01		-7,580570E-01		-6,488401E-02		5,126754E-01
	2,350815E-01		-1,977943E-01		-6,249015E-01		-1,432173E-01
	1,020000E-01		-4,608177E-01		-6,229066E-01		-9,311136E-01
	-2,286546E-01		-3,294397E-01		-2,007141E-01		4,151585E-01
	-2,031270E-01		-1,324977E-01		-3,615781E-01		-5,113091E-01
	3,660114E-02		8,076927E-02		-4,294490E-02		2,470299E-01
	1,052645E-01		1,767865E-01		5,829020E-02		-2,545809E-01
	5,794314E-02		-6,438843E-02		-1,676822E-01		2,852133E-01

Pitch range	C_{13} thru C_{22}	Pitch range	C_{13} thru C_{22}	Pitch range	C_{13} thru C_{22}	Pitch range	C_{13} thru C_{22}
	1,231015E-02		4,720314E-02		-2,835958E-02		-3,163735E-01
	5,452712E-01		1,374098E-01		1,855993E-03		5,261727E-03
	-1,055032E-01		-7,449016E-01		-7,255470E-02		3,512834E-01
	4,865441E-01		-2,392862E-01		-5,444391E-01		-8,774231E-02
	2,485621E-01		-5,064320E-01		-6,373079E-01		-9,555687E-01
	-5,461378E-02		-3,117688E-01		-1,090011E-01		3,227309E-01
	1,175498E-01		-1,233215E-01		-3,698447E-01		-4,916627E-01
	2,772616E-01		-3,754115E-02		-2,054560E-02		2,737616E-01
	8,069777E-02		1,791965E-01		9,060347E-02		-1,856412E-01
	2,700741E-01		-7,785097E-02		-1,565619E-01		2,882749E-01
	1,405139E-01		8,541753E-02		-4,632305E-02		-2,877197E-01
							2,621901E-02
							-2,024138E-01
							-3,691150E-01
							-4,738192E-01
							-3,296576E-01
							-2,743398E-01
							-1,026351E-01
							-1,362258E-01
							-7,479050E-02
							-1,117859E-01

8.2.5.2 Solving front-end equation

The inputs for the SFEQ block are the MFCC vector *C*, an array $HA=\{H_k, k=1,\ldots,N_h\}$ of harmonics and a boolean indicator *voiced_flag*. If current frame is of fully-voiced class then *VH* array is fed into the block (*HA=VH*) and the indicator is set to *voiced_flag = TRUE*. If current frame is of unvoiced class then *UH* array is passed to the block (*HA=UH*) and the indicator is set to *voiced_flag = FALSE*. If the frame is of mixed-voiced class then the block is entered twice, one time with (*HA=VH, voiced_flag=TRUE*) and another time with (*HA=UH, voiced_flag=FALSE*). The SFEQ block outputs an estimate $A^E = \{A_k^E, k = 1,...,N_h\}$ of harmonic magnitudes.

A sequence of processing steps is carried out as described below.

Step 1. Original bins calculation

23-dimensional Inverse Discrete Cosine Transform (IDCT) followed by the exponent operation is applied to the low order cepstra vector $LOC = \{C_k, k=0,\ldots,12\}$ resulting in an *original bins vector* $B^{org} = \{b_k^{org}, k = 1,...,23\}$

$$b_k^{org} = \exp\left(\frac{2}{23}\sum_{n=0}^{12} C_n \times \cos\left(\frac{\pi}{23}\times n\times(k-0{,}5)\right)\right) \tag{8.6}$$

Step 2. Basis vectors calculation

For each harmonic, the (normalized) frequency f_k value is converted to the nearest FFT index $fidx_k$

$$fidx_k = round(f_k \times FFTL), k = 1, N_h \tag{8.7}$$

A binary *grid* vector $G = \{g_n, n = 0,...,FFTL/2\}$ is computed in two steps:

1) $$g_n = 0,\ n=0,\ldots,FFTL/2 \tag{8.8a}$$

2) $$g_{fidx_k} = 1, k = 1,...,N_h \tag{8.8b}$$

23 *prototype basis vectors* PBV_k, $k=1,23$, are calculated. A prototype basis vector $PBV_k = \{pbv_i^k, i = 0,...,FFTL/2\}$ is derived from the triangular window associated with *k*-th frequency channel of the Mel-filters bank described in clause 4.2.9.

$$pbv_i^k = g_i \times \left(0.4 \times \mu_i^k + 0{,}6 \times \mu_i^{k\,2}\right)$$

$$where\ \mu_i^k = \begin{cases} 0,\ if\ \ i < cbin_{k-1}\ and\ i > cbin_{k+1} \\ \dfrac{i - cbin_{k-1} + 1}{cbin_k - cbin_{k-1} + 1},\ if\ \ cbin_{k-1} \le i \le cbin_k \\ 1 - \dfrac{i - cbin_k}{cbin_{k+1} - cbin_k + 1},\ if\ \ cbin_k + 1 \le i \le cbin_{k+1} \end{cases} \quad (8.9)$$

$cbin_k$, $k=0,24$ are defined by (4.6) and (4.8).

(Note that in k-th prototype basis vector only coordinates $pbv^k_{fidx_n}, n = 1,..., N_h$ may have non-zero values.) A *basis vector* $BV_k = \{bv, n = 1,..., N_h\}$ is derived from each prototype basis vector PBV_k by selecting only those coordinates having the indexes $fidx_n$ as follows:

$$BV_k = \{bv_n^k = pbv^k_{fidx_n}, n = 1,..., N_h\},\ k = 1,...,23 \quad (8.10)$$

Step 3. Basis bin vectors and matrix calculation

Each basis vector BV_k is converted to a (in general) complex valued vector $LS_k = \{ls_i^k, i = 0,..., N_h\}$ as specified by the following pseudo code:

```
{
LS_k = BV_k;
    if (voiced_flag == FALSE)
        ls_n^k = bv_n^k · exp(j × φ_n) × peph(f_n)), n = 1,..., N_h ;
}
```

where:

φ_n is a phase associated with n-th unvoiced harmonic as described in clause 8.2.4.2 and $peph$ is phase frequency characteristic of the preemphasis operator:

$$peph(f) = \frac{1 - PE\cos(2\pi \times f) + j \times PE\sin(2\pi \times f)}{\sqrt{1 - 2PE\cos(2\pi \times f) + PE^2}},\quad PE = 0{,}97$$

Note that if *voiced_flag* is TRUE the coordinates of the LS-vectors have real values.

Each LS_k vector is further converted to a synthetic magnitude spectrum vector $SM_k = \{sm_i^k, i = 0,..., FFTL/2 - 1\}$ by convolution with Fourier transformed Hamming window function followed by absolute value operation as follows:

$$sm_i^k = \left| \sum_{n=1}^{N_h} ls_n^k \times HWT(f_n - i/FFTL) \right| \quad (8.11)$$

where:

$$HWT(f) = \begin{cases} 0{,}54\Delta(f) + 0{,}23 \times \left[\Delta\left(f - \dfrac{1}{N-1}\right) + \Delta\left(f + \dfrac{1}{N-1}\right)\right],\ if\ |f| \le WT_BW, \\ 0,\ if\ \ |f| > WT_BW \end{cases} \quad (8.12)$$

$$\Delta(f) = \begin{cases} 0,\ if\ \ f = 0 \\ \sin(\pi \times f \times N)/\sin(\pi \times f) \end{cases} \quad (8.13)$$

$$WT_BW = 0{,}1/f_s \quad (8.14)$$

N is frame length specified in table 4.1, and f_s is sampling rate in kHz.

Mel-filtering operation given by formula (4.7) is applied to each synthetic magnitude spectrum vector SM_k, (in (4.7) bin_i is substituted by sm_i^k), and a 23-dimensional *basis bins vector* $BB_k = \{bb_i^k, i = 1,...,23\}^T$ is obtained. We see the basis bins vectors as column vectors.

A 23-by-23 *basis bins matrix BB* which has the vectors BB_k as its columns is constructed:

$$BB = \begin{bmatrix} BB_1 & BB_2 & ... & BB_{23} \end{bmatrix} \tag{8.15}$$

Step 4. Equation matrix calculation

A 23-by-23 symmetric *equation matrix EM* is computed as follows:

$$EM = BB^T \times BB + 0{,}001 \times \lambda \times E \tag{8.16}$$

where $\lambda = sum(diag(BB^T \times BB))/23$ is an average of the main diagonal elements of the matrix BB^TBB and E is unit 23 by 23 matrix.

In order to reduce the computational complexity of the further processing in the reference implementation, the LU-decomposition is applied to the equation matrix EM, and the LU representation is stored.

Step 5. Initialization of iterative process

Iteration counter is set:

$$it_count = 1$$

Step 6. High bins calculation

This step is carried out only if *voiced_flag* = TRUE, and is skipped otherwise.

23-dimensional IDCT followed by the exponent operation is applied to the high order cepstra vector $HOC = \{C_k, k=13,\ldots,22\}$ output from the HOCR block. The transform results in a *high bins vector* $B^{high} = \{b_k^{high}, k = 1,...,23\}$

$$b_k^{high} = \exp\left(\frac{2}{23}\sum_{n=13}^{22} C_n \times \cos\left(\frac{\pi}{23} \times n \times (k - 0{,}5)\right)\right) \tag{8.17}$$

Step 7. Reference bins calculation

A 23-dimensional reference bins vector $B^{ref} = \{b_k^{ref}, k = 1,...,23\}$ is computed as follows.

```
if (voiced_flag == TRUE)
{
    /* coordinatewise multiplication of Borg and Bhigh vectors */
```

$$b_k^{ref} = b_k^{hig} \times b_k^{org}, k = 1,...,23;$$

```
}
else
{
    /* Borg is taken as Bref */
    Bref = Borg ;
}
```

Step 8. Basis coefficients calculation

A *right side vector* is computed by multiplication of the transposed basis bins matrix by the reference bins vector:

$$V = BB^T \times B^{ref} \tag{8.18}$$

A set of linear equations specified in matrix notation as:

$$EM \times \gamma = V \tag{8.19}$$

is solved and a *basis coefficients vector* $\gamma = \{\gamma_k, k = 1,...,23\}^T$ is obtained:

$$\gamma = EM^{-1} \times V \tag{8.20}$$

In the reference implementation the equations (8.19) are solved using the LU-decomposition representation of the *EM* matrix computed at step 4.

Negative basis coefficients if any are replaced by zero:

$$\gamma_k = \max(0, \gamma_k), k = 1,...,23 \tag{8.21a}$$

Step 9. Control branching

The control branching step is described by the following pseudo code:

```
if (voiced_flag == FALSE  OR  it_count == 3)
{
    go to step 12;
}
/* Otherwise the processing proceeds with the next step 10. */
```

Step 10. Output bins calculation

First, an *output bins vector* $B^{out} = \{b_k^{out}, k = 1,...,23\}^T$ is calculated by the multiplication of the transposed basis bins matrix with the basis coefficients vector:

$$B^{out} = BB^T \times \gamma \tag{8.21b}$$

Then each zero-valued coordinate of this vector (if any) is replaced by a regularization value:

$$\eta = 0{,}005 \times \sum_{k=1}^{23} b_k^{out} \Big/ 23 \tag{8.22}$$

as shown by the following pseudo code instructions being performed for *k=1,…,23*:

```
if
    (b_k^out == 0)
    b_k^out = η;
```

Step 11. High order cepstra refinement

Truncated logarithm operation described in clause 4.2.10 is applied to the coordinates of the output bins vector:

$$lB^{out} = \{lb_k = \max(-50, \ln b_k^{out}), k = 1,...,23\} \tag{8.23}$$

Discrete Cosine Transform (DCT) is applied to the lB^{out} vector, besides only 10 last values are calculated out of 23:

$$C_k^{out} = \sum_{i=1}^{23} lb_i \cos\left(\frac{\pi \times k}{23} \times (i - 0{,}5)\right),\ k = 13,...,22 \tag{8.24}$$

which are considered as new estimate of the High Order Cepstra (HOC). Current high order cepstra values are replaced by these ten coefficients:

$$HOC = \{C_k^{out}, k = 13,...,22\} \tag{8.25}$$

The iteration counter *it_count* is incremented and control is passed to step 6.

Step 12. Harmonic magnitude estimates calculation

The vector $A^E = \{A_k^E, k = 1,...,N_h\}$ of harmonic magnitude estimates is computed as a linear combination of the basis vectors (computed at step 2) weighted by the basis coefficients (computed at step 8):

$$A^E = \sum_{n=1}^{N_h} \gamma_n \times BV_n \tag{8.26a}$$

Finally, the obtained vector is modified in order to cancel the effect of the high frequency preemphasis done in the front-end:

$$\begin{aligned} &A_k^E = A_k^E / MagPemp_k,\ k = 1,...,N_h \\ &where\ MagPemp_k = \sqrt{1 + 0{,}97^2 - 2 \times 0{,}97 \cos(2\pi \times f_k)} \end{aligned} \tag{8.26b}$$

(f_k are harmonic normalized frequencies)

8.2.5.3 Cepstra to magnitudes transformation

From the pitch period and voicing class parameters, the harmonic frequencies f_k, $k=1,\ldots,N_v$ for voiced frames and f_k, $k=1,\ldots,N_u$ for unvoiced frames are computed in clause 8.2.3. One method to estimate the magnitudes at these frequencies from the mel-frequency cepstral coefficients C_0, $C_1,\ldots, C_{12}$ is described in clause 8.2.5.2. In this clause, a second method for transforming cepstra to magnitudes is specified.

As a first step, the high order cepstra are recovered as described in clause 8.2.5.1 for voiced frames to form the complete cepstra C_0, $C_1,\ldots, C_{22}$. For unvoiced frames, the high order cepstra are not recovered. From the cepstra of each frame, a fixed cepstra are subtracted as follows: $D_i = C_i - F_i$, $i = 0, 1,\ldots, 12$ for unvoiced frames and $i = 0, 1,\ldots, 22$ for voiced frames. The fixed Cepstral values F_i are shown in table 8.4. The modified cepstra D_i, $i = 0, 1,\ldots, 12$ (or 22) are used in the estimation of the harmonic magnitudes as described below. To estimate the harmonic magnitude A^I_k at harmonic frequency f_k, the harmonic frequency f_k is first transformed to a corresponding mel-frequency m_k using equation (4.6a) as follows:

$$m_k = 2\,595 \times \log_{10}\left(1 + \frac{f_k}{700}\right) \tag{8.27}$$

The mel-frequency m_k is then transformed to an index j_k with the help of table 8.5. In the table, (integer) index values from 0 to 24 and corresponding mel-frequencies are shown for different sampling rates. Let the mel-frequencies in an appropriate column (e.g. 8 kHz) of table 8.5 be denoted by $M_0,\ldots,M_J,\ldots,M_{24}$. Given a harmonic mel-frequency m_k, it is first bounded so that it does not exceed M_{24}. Then, the index J (in the range from 1 to 24) is found such that $m_k \le M_J$. The (possibly non-integer) index value j_k corresponding to m_k is then calculated as

$$j_k = M_{J-1} + ((m_k - M_{J-1})/(M_J - M_{J-1})) \tag{8.28}$$

From the index j_k, another index l_k is computed as follows:

$$l_k = \begin{cases} 0{,}5; & \text{if } j_k < 0{,}5 \\ 23{,}5; & \text{if } j_k > 23{,}5 \\ j_k; & \text{otherwise} \end{cases} \tag{8.29}$$

From the modified cepstra D_i, $i = 0, 1,\ldots, 12$ (or 22), and the index l_k, the log-magnitude estimate a_k is obtained as

$$a_k = \frac{D_0}{23} + \frac{2}{23} \sum_{i=1}^{Max_i} D_i \cos((l_k - 0{,}5) \times i \times (\pi/23)) \tag{8.30}$$

where, *Max_i* is 12 or 22 depending on whether the frame is unvoiced or voiced respectively. From a_k, the harmonic magnitude estimate A^I_k is obtained as follows:

$$A^I{}_k = \begin{cases} \exp(a_k)\times 2\times(m_k/(M_0+M_1)); & \text{if } j_k < 0{,}5 \\ \exp(a_k)\times 2\times(24-j_k); & \text{if } j_k > 23{,}5 \\ \exp(a_k); & \text{otherwise} \end{cases} \quad (8.31a)$$

The above method (8.27 through 8.31) is applied to each harmonic frequency to estimate the harmonic magnitudes A^I_k for k = 1, 2,…, N_u (or N_v).

Table 8.4: Fixed cepstral values

Fixed Cepstral values F_0 through F_{22}		
8 kHz	**11 kHz**	**16 kHz**
3,2536754e+01	2,9787007e+01	3,9470548e+01
-1,9789891e+01	-2,1651108e+01	-2,4909673e+01
-3,0452398e+00	-2,8123724e+00	-2,8222240e+00
-2,8438349e+00	-2,8365281e+00	-3,2486815e+00
-1,1122951e+00	-8,3974459e-01	-1,0130151e+00
-1,0977202e+00	-1,1305221e+00	-1,2072917e+00
-4,6039646e-01	-4,8309548e-01	-4,9382294e-01
-6,1778432e-01	-3,9550496e-01	-5,3244176e-01
-5,1890050e-01	-4,8029016e-01	-9,2284267e-02
-6,1115379e-01	-4,9048922e-01	-1,1890436e-01
-7,3124391e-02	-1,2010290e-01	-9,3826506e-02
-1,6289170e-01	-8,2625448e-02	-8,3366636e-02
-2,6278086e-01	-4,8331334e-02	-9,8590481e-03
-8,9079853e-02	-1,6097046e-01	-2,8945789e-01
-2,3033581e-01	-3,8010135e-01	-1,0346320e-01
8,0049444e-02	-4,5318985e-01	-1,3629115e-01
1,7049236e-01	-8,7807383e-02	-2,5264593e-01
8,3038678e-02	-7,4268073e-02	-2,1191457e-01
8,1462604e-02	1,4095451e-01	-4,8705782e-02
2,3817306e-02	1,0216903e-01	-1,0828508e-02
6,2334655e-03	-1,0476986e-02	7,7213331e-02
-8,0701593e-02	-4,6905935e-02	4,2159844e-02
-2,4373608e-02	-4,0755373e-02	5,3725857e-03

Table 8.5: Index values and corresponding mel-frequencies

Index value	Mel Frequencies		
	8 kHz	11 kHz	16 kHz
0	9,6383e+01	6,7139e+01	9,6383e+01
1	1,8517e+02	1,9049e+02	2,2707e+02
2	2,6747e+02	3,0167e+02	3,4416e+02
3	3,4416e+02	4,0285e+02	4,5023e+02
4	4,5023e+02	4,9569e+02	5,4716e+02
5	5,1577e+02	5,8147e+02	6,6467e+02
6	6,0745e+02	6,9901e+02	7,9617e+02
7	6,9222e+02	7,7107e+02	8,9133e+02
8	7,7107e+02	8,7119e+02	1,0205e+03
9	8,6828e+02	9,9219e+02	1,1179e+03
10	9,5777e+02	1,0751e+03	1,2415e+03
11	1,0407e+03	1,1770e+03	1,3528e+03
12	1,1179e+03	1,2704e+03	1,4679e+03
13	1,2075e+03	1,3772e+03	1,5848e+03
14	1,2906e+03	1,4748e+03	1,7018e+03
15	1,3827e+03	1,5817e+03	1,8078e+03
16	1,4679e+03	1,6793e+03	1,9231e+03
17	1,5472e+03	1,7692e+03	2,0442e+03
18	1,6330e+03	1,8657e+03	2,1535e+03
19	1,7238e+03	1,9667e+03	2,2668e+03
20	1,8078e+03	2,0595e+03	2,3819e+03
21	1,8859e+03	2,1656e+03	2,4973e+03
22	1,9766e+03	2,2625e+03	2,6120e+03
23	2,0605e+03	2,3604e+03	2,7251e+03
24	2,1461e+03	2,4582e+03	2,8400e+03

8.2.5.4 Combined magnitudes estimate calculation

This block calculates a final combined estimate $A = \{An, n = 1,..., N_h)$ of harmonic magnitudes from the estimates $A^E = \{A_n^E, n = 1,..., N_h\}$ and $A^I = \{A_n^I, n = 1,..., N_h\}$ obtained in SFEQ block (clause 8.2.5.2) and CTM block (clause 8.2.5.3) correspondingly. Voiced and unvoiced harmonic arrays are treated slightly differently.

8.2.5.4.1 Combined magnitude estimate for unvoiced harmonics

Vector A^E is scaled so that its squared norm is equal to the squared norm of the A^I vector as is specified by the pseudo code:

```
{
    E^E = sum_{n=1}^{N_h} A_n^{E 2};
    if (EE == 0)
        sc = 0;
    else
    {
        E^I = sum_{n=1}^{N_h} A_n^{I 2};
        sc = sqrt(E^I / E^E);
        A^E = sc × A^I;
    }
}
```

The magnitudes AE and AI are mixed:

$$A = 0{,}9 \times A^E + 0{,}1 \times A^I$$

8.2.5.4.2 Combined magnitude estimate for voiced harmonics

Vector A^E is scaled and then mixed with the A^I vector using a pitch dependent mixing proportion. Pitch period value p_{fixed} measured in 8kHz samples is used further.

Scaling

Scaling is performed differently for long and short pitch period values.

If p_{fixed} value is less than 55 samples then A^E vector is scaled exactly as is described in clause 8.2.5.4.1. Otherwise (if $p_{fixed} \geq 55$) the scaling procedure described below is carried out.

Two scaling factors sc_{low} and sc_{high} are calculated in frequency bands [0, 1 200 Hz] and [1 200 Hz,$F_{Nyquist}$] respectively.

$$sc_{low} = \sqrt{\frac{\sum_{n=1}^{L} A_n^{I\,2}}{\sum_{n=1}^{L} A_n^{E\,2}}}, \qquad sc_{high} = \sqrt{\frac{\sum_{n=L+1}^{Nh} A_n^{I\,2}}{\sum_{n=L+1}^{Nh} A_n^{E\,2}}}$$

where:

$$L = floor(1\,200 \times p / (1\,000 \times f_s)),$$

p is the pitch period in samples corresponding to the actual sampling rate f_s kHz. A scaling factor is set to 0 if the denominator of the corresponding expression is equal to zero.

Then the harmonic magnitudes A_n^E are modified as specified by the following pseudo code section being executed for $n=1,\ldots,H_h$.

```
{
    fHz = f_n × f_s × 1 000 ; /* harmonic frequency in Hz units */
    if ( fHz ≤ 200 )
        A_n^E = A_n^E × sc_low ;
    elseif ( fHz ≥ 2 500
        A_n^E = A_n^E × sc_high ;
    else
    {
        λ = (2 500 − fHz)/(2 500 − 200) ;
        sc = λ × sc_low + (1 − λ) × sc_high ;
        A_n^E = A_n^E × sc ;
    }
}
```

Mixing.

Mixture parameter values χ_n as a function of $p_{fixed\,n}$ values are specified by table 8.6.

Table 8.6: Magnitude mixture parameter *vs*. pitch

N	$p_{fixed\,n}$	χ_n	n	$p_{fixed\,n}$	χ_n
1	22,5	0,3387	14	87,5	0,8556
2	27,5	0,3556	15	92,5	0,8773
3	32,5	0,4130	16	97,5	0,9144
4	37,5	0,4908	17	102,5	0,9098
5	42,5	0,5952	18	107,5	0,8756
6	47,5	0,7365	19	112,5	0,8007
7	52,5	0,8492	20	117,5	0,7109
8	57,5	0,8904	21	122,5	0,6571
9	62,5	0,8332	22	127,5	0,6389
10	67,5	0,7679	23	132,5	0,6291
11	72,5	0,7475	24	137,5	0,5768
12	77,5	0,7921	25	142,5	0,5231
12	82,5	0,8227	26	147,5	0,5231

The mixture parameter value χ to be used for mixing the magnitude vectors is determined by linear interpolation between the values given by the table as described by the following pseudo code:

```
{
    if ( p_fixed ≤ p_fixed1 )
        χ = χ_1 ;
    elseif ( p_fixed ≥ p_fixed26 )
        χ = χ_26 ;
    else
    {
        Find n such that  p_fixedn ≤ p_fixed < p_fixedn+1 ;
        ρ = (p_fixedn+1 − p_fixed)/(p_fixedn+1 − p_fixedn) ;
        χ = ρ × χ_n + (1 − ρ) × χ_n+1 ;
    }
    A = χ × A^E + (1 − χ) × A^I ;
}
```

8.2.6 All-pole spectral envelope modelling

Given the harmonic magnitudes estimate, A_k, k = 1, 2,…, N_v , of a voiced frame, an all-pole model is derived from the magnitudes as specified in this clause. The all-pole model parameters a_j, j = 1, 2,…, J are used for postfiltering (clause 8.2.7) and harmonic phase synthesis (clause 8.2.8). The model order J is 10 for 8 kHz, 14 for 11 kHz, and 18 for 16 kHz sampling frequency respectively.

The magnitudes are first normalized as specified by the pseudo-code below so that the largest normalized value is 1.

```
if (max(A_k) > 0)
    B_k = A_k / max(A_k); k = 1, 2,…, N_v
else
    a_j = 0;  j = 1, 2,…, J
```

From the normalized magnitudes, a set of interpolated magnitudes is derived. The size of the interpolated vector is given by $K = (N_v - 1) \times F + 1$, where the interpolation factor F is a function of N_v as shown in table 8.7. The interpolated vector is obtained by introducing $(F - 1)$ additional magnitudes through linear interpolation between each consecutive pair of the original magnitudes. When $F = 1$, i.e. when $N_v \geq 25$, there is no interpolation and $K = N_v$. The interpolated vector is specified as follows:

$$G_k = \begin{cases} B_{1+(k-1)/F}; & k = 1, F+1, 2F+1, ..., (N_v-1)F+1 \\ B_j + \dfrac{k-(j-1)F-1}{F}(B_{j+1} - B_j); & (j-1)F+1 < k < jF+1, \quad j = 1,2,...,N_v-1 \end{cases} \tag{8.31b}$$

where, G_k, k = 1, 2,…, $K = (N_v - 1) \times F + 1$ represent the interpolated magnitude vector. Each of the interpolated magnitudes is then assigned a normalized frequency in the range from 0 to π, viz., $\omega_k = k \times \pi / (K+1)$, k = 1, 2,…, K. The interpolated vector is next augmented by two additional magnitude values corresponding to $\omega_k = 0$ (DC) and $\omega_k = \pi$. The length of the augmented, interpolated vector is thus $K + 2$. This vector is still denoted by G_k, but the subscript k now ranges from 0 to $K + 1 = (N_v - 1) \times F + 2$. The values of G_0 and G_{K+1} are obtained as shown in the pseudo-code below.

```
if (F == 1)
    {
        G_K+1 = G_K;
if (G_2 > 1.2 G_1)
        G_0 = 0.8 G_1;
    else if (G_2 < 0.8 G_1)
        G_0 = 1.2 G_1;
    else
        G_0 = G_1;
}
else
{
    G_K+1 = 2.0 (G_K - G_K-1);
    G_0 = 2.0 (G_1 - G_2);
}
```

Table 8.7: Interpolation factor *vs*. number of harmonics

Number of voiced harmonics	Interpolation factor
$N_V < 12$	4
$12 \le N_V < 16$	3
$16 \le N_V < 25$	2
$25 \le N_V$	1

From the augmented, interpolated vector G_k, k = 0, 1,…, K+1, a pseudo-autocorrelation function R_j is computed using the cosine transform as follows:

$$R_j = G_0 + (-1)^j G_{K+1} + 2\sum_{k=1}^{K} G_i \cos(\omega_k \cdot j);\ j = 0, 1, \ldots, J \tag{8.32}$$

From the pseudo-autocorrelation coefficients R_j, j = 0, 1,…, J, the all-pole model parameters a_j, j = 1, 2,…, J are obtained through the well known Levinson-Durbin recursion as the solution of the normal equations:

$$\sum_{j=1}^{J} a_j \times R_{i-j} = R_i;\ 1 \le i \le J \tag{8.33}$$

For the case when F = 1, i.e. when $N_v \ge 25$, the all-pole model parameters derived as above represent the final values. For other cases when $F > 1$, the model parameters are further refined as specified below. The spectral envelope defined by the all-pole model parameters is given by:

$$H(\omega) = \frac{1}{\left|1 + a_1 e^{-j\omega} + a_2 e^{-j2\omega} + \ldots + a_J e^{-jJ\omega}\right|^2} \tag{8.34}$$

where, the $e^{j\omega}$ represents a complex exponential at frequency ω. The spectral envelope given by (8.32) is sampled at all the frequencies $\omega_k = k\pi / (K+1)$, k = 0, 1, ..., K+1 to obtain the modelled magnitudes H_k, k = 0, 1,…, K+1. The maximum of the modelled magnitudes at frequencies corresponding to the original estimated magnitudes is then used to normalize the modelled magnitudes as follows:

$$L_k = H_k \Big/ \max(H_k \ \big|\ k = 1, F+1, 2F+1, \ldots, (N_v - 1)F + 1);\quad k = 0, 1, \ldots,\ K+1 \tag{8.35}$$

Next, scale factors S_k , k = 0, 1,…, K + 1 are computed as follows:

$$S_k = \begin{cases} 1; & k = 0 \text{ and } k = K+1 \\ G_k / L_k; & k = 1, F+1, 2F+1, ..., (N_v - 1)F + 1 \\ S_{(j-1)F+1} + \dfrac{k-(j-1)F-1}{F}(S_{jF+1} - S_{(j-1)F+1}); & (j-1)F+1 < k < jF+1; \quad j = 1, 2, ..., (N_v - 1) \end{cases} \tag{8.36}$$

The normalized, modelled magnitudes are then multiplied by the appropriate scale factors to obtain a new set of magnitudes $M_k = L_k S_k$, k = 0, 1, …, K+1. This set of magnitudes is used to compute a new pseudo-autocorrelation function using (8.32) and subsequently a new set of all-pole model parameters as a solution (8.33) as the final values.

8.2.7 Postfiltering

Postfiltering is applied to the harmonic magnitudes A_k, k = 1, 2,…, N_v of a voiced frame to emphasize the formants in the speech signal using the all-pole model parameters a_j, j = 1, 2,…, J as specified below.

From the number of voiced harmonics N_v, the interpolation factor F from table 8.7 and the interpolated vector size $K = (N_v - 1)\,F + 1$ are first determined. Then, a weighting factor U_k is computed for each harmonic as follows:

$$\theta_k = ((k-1) \times F + 1) \times (\pi / (K+1)) \tag{8.37a}$$

$$\mathrm{Re}1_k = 1 + \sum_{j=1}^{J} a_j \times \alpha^j \times \cos(j \times \theta_k)\,;\ k = 1, 2, \ldots, N_v \tag{8.38b}$$

$$\mathrm{Im}1_k = -\sum_{j=1}^{J} a_j \times \alpha^j \times \sin(j \times \theta_k)\,;\ k = 1, 2, \ldots, N_v \tag{8.39c}$$

$$\mathrm{Re}2_k = 1 + \sum_{j=1}^{J} a_j \times \beta^j \times \cos(j \times \theta_k)\,;\ k = 1, 2, \ldots, N_v \tag{8.40d}$$

$$\mathrm{Im}2_k = -\sum_{j=1}^{J} a_j \times \beta^j \times \sin(j \times \theta_k)\,;\ k = 1, 2, \ldots, N_v \tag{8.41e}$$

$$\mathrm{Re}3_k = 1 - \mu \times \cos(\theta_k)\,;\ k = 1, 2, \ldots, N_v \tag{8.42f}$$

$$\mathrm{Im}3_k = \mu \times \sin(\theta_k)\,;\ k = 1, 2, \ldots, N_v \tag{8.43g}$$

$$U_k = \frac{[(\mathrm{Re}2_k)^2 + (\mathrm{Im}2_k)^2] \times \sqrt{[(\mathrm{Re}3_k)^2 + (\mathrm{Im}3_k)^2]}}{[(\mathrm{Re}1_k)^2 + (\mathrm{Im}1_k)^2]}\,;\ k = 1, 2, \ldots, N_v \tag{8.44h}$$

The values of α, β, and μ are respectively 0,95, 0,75 and 0,5. The weights are then normalized and bounded as follows.

$$V_k = U_k \Big/ \left(\frac{1}{N_v} \sum_{k=1}^{N_v} {U_k}^4 \right)^{0.25}\,;\ k = 1, 2, \ldots, N_v \tag{8.45a}$$

$$W_k = \max(0{,}5, \min(1{,}5, V_k))\,;\ k = 1, 2, \ldots, N_v \tag{8.45b}$$

Postfiltering is applied to the harmonic magnitudes as follows. It is ensured that the energy in the harmonics before and after postfiltering remains the same.

$$B_k = \begin{cases} A_k \times W_k\,; & \textit{if } \theta_k \geq 0{,}05 \times \pi \\ A_k\,; & \textit{otherwise} \end{cases} \tag{8.46a}$$

$$\rho = \sqrt{\sum_{k=1}^{N_v} W_k^{\,2} \Big/ \sum_{k=1}^{N_v} B_k^{\,2}} \tag{8.46b}$$

$$P_k = \rho \times B_k \,;\; k = 1, 2, \ldots, N_v \tag{8.47c}$$

where P_k, $k = 1, 2, \ldots, N_v$ represent the postfiltered harmonic magnitudes.

8.2.8 Voiced phase synthesis

The harmonic phases φ_k, $k = 1, 2, \ldots, N_v$ of a voiced frame with harmonic cyclic frequencies $\omega_k = 2\pi f_k$, $k = 1, 2, \ldots, N_v$ are specified as follows. Each harmonic phase φ_k is made up of three components: a linear phase component $\varphi_{k,lin}$, an excitation phase component $\varphi_{k,exc}$, and an envelope phase component $\varphi_{k,env}$.

The linear phase component is computed as follows:

$$\varphi_{k,lin} = \begin{cases} 0; & if\ previous\ frame\ is\ unvoiced \\ (\varphi_{1,lin,prev} \times RF + \omega_{1,ave} \times M) \times k; & otherwise \end{cases} \tag{8.48}$$

where:

- $\varphi_{1,lin,prev}$ represents the linear phase component of the fundamental phase of the previous frame;

RF represents a rational factor of the *R1/R2*, where $R1, R2 \in \{1, 2, 3, 4\}$, such that the jump given by $\left| p \times R1 - p_{prev} \times R2 \right| \Big/ p \times R1$ between the previous pitch period (p_{prev}) and current pitch period (p) is minimized;

$\omega_{1,ave}$ is the weighted sum of the fundamental (cyclic) frequency of the previous and current frames given by $\omega_{1,ave} = (\omega_{1,prev} \times RF + \omega_1)/2$, and M is the frame shift in samples.

Note that p_{prev} and $\varphi_{1,lin,prev}$ are initialized to 0 (meaning the previous frame is unvoiced) when the very first frame is being processed.

The excitation phase component is determined using table 8.8 as follows. Given a harmonic frequency ω_k, it is first transformed into an integer index $I_k = round(256 \times \omega_k / \pi)$, the corresponding value $T[I_k]$ from table 8.7 is looked up, and un-normalized to obtain $\varphi_{k,exc} = T[I_k]\,\pi$.

The envelope phase component is computed using the all-pole model parameters, a_j, $j = 1, 2, \ldots, J$, as follows. From the number of voiced harmonics N_v, the interpolation factor F from table 8.7 and the interpolated vector size $K = (N_v - 1)\,F + 1$ are first determined. Then the envelope phase component is computed as:

$$\theta_k = ((k-1) \times F + 1) \times (\pi / (K+1)) \,;\; k = 1, 2, \ldots, N_v \tag{8.49a}$$

$$\mathrm{Re}_k = 1 + \sum_{j=1}^{J} a_j \cos(j \times \theta_k) \,;\; k = 1, 2, \ldots, N_v \tag{8.49b}$$

$$\mathrm{Im}_k = -\sum_{j=1}^{J} a_j \sin(j \times \theta_k) \,;\; k = 1, 2, \ldots, N_v \tag{8.49c}$$

$$\varphi_{k,env} = (-2) \times \tan^{-1}(\mathrm{Im}_k / \mathrm{Re}_k) \,;\; k = 1, 2, \ldots, N_v \tag{8.49d}$$

The excitation and envelope components of the phases are added and any linear component is removed as follows:

$$\varphi_{k,sum} = \varphi_{k,exc} + \varphi_{k,env};\quad k = 1, 2, \ldots, N_v \tag{8.50a}$$

$$\varphi_{k,add} = \varphi_{k,sum} - k \cdot \varphi_{1,sum};\quad k = 1, 2, \ldots, N_v \tag{8.50b}$$

The linear phase component and the additional phase component are added to obtain the harmonic phases for the voiced frame as follows:

$$\varphi_k = \varphi_{k,lin} + \varphi_{k,add}; \quad k = 1, 2, ..., N_v \tag{8.51}$$

Table 8.8: Normalized excitation phases

Index	Normalized phase	Index	Normalized phase	Index	Normalized phase	Index	Normalized phase
0	0,000000	64	0,806122	128	-0,428986	192	0,231750
1	0,577271	65	0,761841	129	-0,449249	193	0,219360
2	0,471039	66	0,707184	130	-0,476257	194	0,211182
3	0,402039	67	0,649353	131	-0,512085	195	0,207703
4	0,341461	68	0,595245	132	-0,555054	196	0,209747
5	0,282104	69	0,553375	133	-0,601379	197	0,215332
6	0,221069	70	0,535004	134	-0,646881	198	0,217590
7	0,157074	71	0,551025	135	-0,687469	199	0,208527
8	0,089905	72	0,593689	136	-0,720123	200	0,184631
9	0,019989	73	0,629669	137	-0,743896	201	0,147583
10	-0,051819	74	0,641205	138	-0,760712	202	0,101593
11	-0,124237	75	0,637146	139	-0,774292	203	0,051697
12	-0,195770	76	0,630432	140	-0,786865	204	0,002960
13	-0,264679	77	0,626068	141	-0,796417	205	-0,039154
14	-0,328705	78	0,618439	142	-0,797058	206	-0,068756
15	-0,385162	79	0,597534	143	-0,782288	207	-0,080597
16	-0,430573	80	0,558716	144	-0,753052	208	-0,073730
17	-0,460846	81	0,504242	145	-0,723755	209	-0,055573
18	-0,472351	82	0,439545	146	-0,710052	210	-0,038666
19	-0,464783	83	0,371796	147	-0,714722	211	-0,030792
20	-0,444977	84	0,314423	148	-0,731720	212	-0,033630
21	-0,425323	85	0,322479	149	-0,753998	213	-0,047180
22	-0,415466	86	0,692352	150	-0,776672	214	-0,072174
23	-0,418579	87	0,820557	151	-0,797760	215	-0,109039
24	-0,433502	88	0,775940	152	-0,817749	216	-0,156860
25	-0,457764	89	0,703735	153	-0,838562	217	-0,213318
26	-0,488617	90	0,625885	154	-0,861664	218	-0,275146
27	-0,523315	91	0,549744	155	-0,887115	219	-0,338562
28	-0,559174	92	0,479889	156	-0,913971	220	-0,398956
29	-0,593689	93	0,420258	157	-0,941437	221	-0,450836
30	-0,625031	94	0,374023	158	-0,969849	222	-0,487793
31	-0,652130	95	0,341888	159	0,999176	223	-0,505707
32	-0,674835	96	0,319366	160	0,963562	224	-0,510162
33	-0,693390	97	0,297546	161	0,922089	225	-0,518524
34	-0,707428	98	0,268768	162	0,875092	226	-0,545410
35	-0,715729	99	0,230896	163	0,824432	227	-0,592499
36	-0,717133	100	0,186066	164	0,773285	228	-0,654510
37	-0,713837	101	0,137939	165	0,726074	229	-0,725586
38	-0,713104	102	0,090027	166	0,688934	230	-0,801025
39	-0,723785	103	0,045288	167	0,669617	231	-0,877136
40	-0,750366	104	0,005859	168	0,674377	232	-0,950897
41	-0,791931	105	-0,026398	169	0,698090	233	0,980316
42	-0,845093	106	-0,049316	170	0,719421	234	0,918762
43	-0,905945	107	-0,059448	171	0,721069	235	0,866211
44	-0,970825	108	-0,052521	172	0,702698	236	0,824219
45	0,963654	109	-0,028687	173	0,671631	237	0,795319
46	0,901123	110	-0,000732	174	0,634674	238	0,786377
47	0,846222	111	0,012024	175	0,596527	239	0,810913
48	0,805481	112	0,001312	176	0,559784	240	0,872406
49	0,788788	113	-0,028900	177	0,525757	241	0,925385
50	0,807312	114	-0,070801	178	0,494995	242	0,926483
51	0,857269	115	-0,117004	179	0,468231	243	0,882111
52	0,904724	116	-0,160583	180	0,446991	244	0,808807
53	0,922668	117	-0,194824	181	0,433105	245	0,716248
54	0,913757	118	-0,214020	182	0,427216	246	0,608063
55	0,888916	119	-0,217743	183	0,426483	247	0,480927
56	0,856750	120	-0,215424	184	0,424225	248	0,310974
57	0,823730	121	-0,221161	185	0,414124	249	-0,054810
58	0,796082	122	-0,241730	186	0,393951	250	-0,554077

Index	Normalized phase	Index	Normalized phase	Index	Normalized phase	Index	Normalized phase
59	0,781250	123	-0,274475	187	0,365723	251	-0,763275
60	0,786346	124	-0,313202	188	0,333374	252	-0,904968
61	0,809631	125	-0,351440	189	0,301086	253	0,977448
62	0,831787	126	-0,384247	190	0,272278	254	0,884125
63	0,831818	127	-0,409363	191	0,249054	255	0,849152
						256	0,999969

8.2.9 Line spectrum to time-domain transformation

This block transforms a line spectrum of the frame represented by an array $H=\{H_n=(f_n,A_n,\varphi_n),n=1,...,N_h\}$ of harmonics to a time-domain speech signal segment. If the frame is of fully-voiced class as indicated by *vc =="fully-voiced"* the array H is set to *VH*. In case of unvoiced frame (*vc =="unvoiced"*) H is set to UH. In the case of mixed-voiced frame the arrays of voiced and unvoiced harmonics are combined as described in the following clause.

8.2.9.1 Mixed-voiced frames processing

This step is performed for the mixed-voiced frames only as indicated by *vc == "mixed_voiced"*. The input to the step are the array $VH=\{H_n^v=(f_n^v,A_n^v,\varphi_n^v),n=1,...,N_v\}$ of voiced harmonics and the array $UH=\{H_n^u=(f_n^u,A_n^u,\varphi_n^u),n=1,...,N_u\}$ of unvoiced harmonics. The output is a combined array $H=\{H_n=(f_n,A_n,\varphi_n),n=1,...,N_h\}$ of harmonics. The combined array contains the voiced harmonics associated with frequencies lower than 1 200 Hz and the unvoiced harmonics associated with frequencies higher than 1 200 Hz. The processing is described by the following pseudo code:

```
{
    v_last = ceil(1 200/(1 000× f_s × p)) ; /* index of the last voiced harmonic to be taken */
    u_first = ceil(1 200/(1 000× f_s × FFTL))+1; /* index of the first unvoiced harmonic to be taken  */
    sc = sqrt( (sum_{n=v_last+1}^{N_v} A_n^{v 2}) / (sum_{n=u_first}^{N_u} A_n^u) ) ; /* compute magnitude scaling factor */
    H_n = H_n^v, n = 1,...,v_last ;
    f_{v_last+n−u_first+1} = f_n^u,   φ_{v_last+n−u_first+1} = φ_n^u, n = u_first,...,N_u ;
    A_{v_last+n−u_first+1} = sc× A_n^u, n = u_first,...,N_u ;
    N_h = v_last + N_u − u_first +1 ;
}
```

8.2.9.2 Filtering very high-frequency harmonics

At this step the harmonics associated with the frequencies close enough to the Nyquist frequency (if any) are filtered out. Those elements of the harmonics array which satisfy the condition:

$$round(f\times FFTL) > round(0{,}93\times FFTL/2)$$

are eliminated and the number N_h of harmonics is updated appropriately.

8.2.9.3 Energy normalization

A synthetic complex discrete spectrum is calculated:

$$sd_i = \sum_{n=1}^{N_h} A_n \times \exp(j \times \varphi_n) \times \Delta(f_n - i/FFTL), i = 0,..., FFTL/2 \quad (8.52)$$

by convolution of the line spectrum with truncated Dirichlet kernel:

$$\Delta(f) = \begin{cases} 0, \ if \quad f = 0 \ or \ |f| > WT_BW \\ \sin(\pi \times f \times N)/\sin(\pi \times f), \ otherwise \end{cases} \quad (8.53)$$

where WT_BW is given by (8.14). Then the frame energy estimate E_e is calculated:

$$E_e = \frac{1}{FFTL}\left(|sd_0|^2 + |sd_{FFTL/2}|^2 + 2 \times \sum_{i=1}^{FFTL/2-1} |sd_i|^2 \right) \quad (8.54)$$

If the energy estimate is nonzero a normalization factor *NF* is computed using the logE parameter extracted from the decoded feature vector:

$$NF = \sqrt{\exp(\log E)/E_e}, \quad (8.55)$$

otherwise the normalization factor is set to zero *NF = 0*.

The harmonic magnitudes are scaled:

$$A_n = NF \times A_n, \ n = 1,..., N_h \quad (8.56)$$

8.2.9.4 STFT spectrum synthesis

A synthetic complex discrete spectrum *s_stft* is calculated like in (8.52) but Fourier transform of *2M* (*M* is frame shift) samples long Hann window is used instead of the Dirichlet kernel:

$$s_stft_i = \sum_{n=1}^{N_h} A_n \times \exp(j \times \varphi_n) \times HnWT(f_n - i/FFTL), i = 0,..., FFTL/2, \quad (8.57)$$

$$HWT(f) = \begin{cases} 0{,}50\Delta(f) + 0{,}25 \times \left[\Delta\left(f - \frac{1}{2M-1} \right) + \Delta\left(f + \frac{1}{2M-1} \right) \right], \ if \ |f| \le WT_BW, \\ 0, \ if \ |f| > WT_BW \end{cases} \quad (8.58)$$

where Δ is given by (8.13), and *WT_BW* by (8.14).

8.2.9.5 Inverse FFT

An inverse FFT is applied to the synthetic STFT spectrum resulting in FFTL-dimensional vector $S_{syn} = \{s_n^{syn}, n = 0,..., FFTL - 1\}$ with real coordinates which is used as a time-domain representation of current frame:

$$s_n^{syn} = \frac{1}{FFTL} \sum_{i=0}^{FFTL-1} s_stft_i \times \exp(j \times i \times n \frac{2\pi}{FFTL}) \quad (8.59)$$

In (8.59) $s_stft_i = s_stft^*_{FFTL-i-1}$ *if* $i \ge FFTL/2$.

8.2.10 Overlap-Add

The input to the Overlap-Add block (OLA) is the synthesized time-domain frame S^{syn}. The OLA block outputs an M samples long segment of speech which is appended to the already synthesized part of the speech signal. The OLA block maintains a pair of M samples long buffers:

$$BUF^{out} = \{buf_k^{out}, k = 1,...,M\}; \text{ and}$$

$$BUF^{ola} = \{buf_k^{ola}, k = 1,...,M\}.$$

Each coordinate of BUF^{ola} is initialized by zero values when the very first frame is processed. BUF^{ola} preserves its contents in between invocations of the OLA block. The procedure performed in the OLA block is specified by the following pseudo code:

```
{
    buf_{k+1}^{ola} = buf_{k+1}^{ola} + s_stft_{FFTL-M+k}, k = 1,...,M-1;   /* overlap-add */
    buf_k^{out} = buf_k^{ola}, k = 1,...,M ;   /* copy OLA buffer to OUT buffer */
    buf_k^{ola} = s_stft_{k-1}, k = 1,...,M ;   /* prepare for the next frame */
    Output BUF^{out} ;
}
```

History

Document history		
V1.1.1	August 2003	Membership Approval Procedure MV 20031024: 2003-08-26 to 2003-10-24