Document: C.S0020-0

Version:

Date: December 1999

3RD GENERATION PARTNERSHIP PROJECT 2 "3GPP2"

High Rate Speech Service Option 17 for Wideband Spread Spectrum Communication Systems

High Rate Speech Service Option 17 for Wideband Spread Spectrum Communication Systems

TIA/EIA/IS-733

Publish Version

November 17, 1997

PREFACE

These technical requirements form a standard for Service Option 17, a variable rate, two-way speech service option. The maximum speech coding rate of the service option is 13.3 kbps.

This standard does not address the quality or reliability of Service Option 17, nor does it cover equipment performance or measurement procedures.

SECTION SUMMARY

1. **General.** This section defines the terms and numeric indicators used in this document.

2. **Service Option 17: Variable Data Rate Two-Way Voice.** This section describes the requirements for Service Option 17. Included in these requirements is the description of a speech codec algorithm for variable rate, two-way voice.

3. **Annex A**. Bibliography. This is an informative annex (not considered part of this standard) listing documents which may be useful in implementing the standard.

NOTES

1. TIA/EIA/IS-736 "Recommended Minimum Performance Standard for the High Rate Speech Service Option for Wideband Spread Spectrum Communication Systems," provides specifications and measurement methods.

2. "Base station" refers to the functions performed on the land side, which are typically distributed among a cell, a sector of a cell, and a mobile switching center.

3. Section 2 uses the following verbal forms: "Shall" and "shall not" identify requirements to be followed strictly to conform to the standard and from which no deviation is permitted. "Should" and "should not" indicate that one of several possibilities is recommended as particularly suitable, without mentioning or excluding others; that a certain course of action is preferred but not necessarily required; or that (in the negative form) a certain possibility or course of action is discouraged but not prohibited. "May" and "need not" indicate a course of action permissible within the limits of the standard. "Can" and "cannot" are used for statements of possibility and capability, whether material, physical, or causal.

4. Footnotes appear at various points in this specification to elaborate and further clarify items discussed in the body of the specification.

5. Unless indicated otherwise, this document presents numbers in decimal form.

   Binary numbers are distinguished in the text by the use of single quotation marks. In some tables, binary values may appear without single quotation marks if table notation clearly specifies that values are binary. The character 'x' is used to represent a binary bit of unspecified value. For example 'xxx00010' represents any 8-bit binary value such that the least significant five bits equal '00010'.

   Hexadecimal numbers (base 16) are distinguished in the text by use of the form 0xh…h where h…h represents a string of hexadecimal digits. For example, 0x2fa1 represents a number whose binary value is '10111110100001' and whose decimal value is 913.

NOTES

6. The following conventions apply to mathematical expressions in this standard:

- $\lfloor x \rfloor$ indicates the largest integer less than or equal to x: $\lfloor 1.1 \rfloor = 1$, $\lfloor 1.0 \rfloor = 1$.
- $\lceil x \rceil$ indicates the smallest integer greater than or equal to x: $\lceil 1.1 \rceil = 2$, $\lceil 2.0 \rceil = 2$.
- $|x|$ indicates the absolute value of x: $|-17|=17$, $|17|=17$.
- $\oplus$ indicates exclusive OR.
- min(x, y) indicates the minimum of x and y.
- max(x, y) indicates the maximum of x and y.
- In figures, $\otimes$ indicates multiplication. In formulas within the text, multiplication is implicit. For example, if h(n) and $p_L(n)$ are functions, then $h(n)\, p_L(n) = h(n) \otimes p_L(n)$.
- x mod y indicates the remainder after dividing x by y: $x \bmod y = x - (y \lfloor x/y \rfloor)$.
- round(x) is traditional rounding: $\mathrm{round}(x) = \lfloor x + 0.5 \rfloor$.

$$\mathrm{sign}(x) = \begin{cases} 1 & x \geq 0 \\ -1 & x < 0 \end{cases}.$$

- $\sum$ indicates summation. If the summation symbol specifies initial and terminal values, and the initial value is greater than the terminal value, then the value of the summation is 0. For example, if N=0, and if f(n) represents an arbitrary function, then

$$\sum_{n=1}^{N} f(n) = 0.$$

- The bracket operator, [ ], isolates individual bits of a binary value. VAR[n] refers to bit n of the binary representation of the value of the variable VAR, such that VAR[0] is the least significant bit of VAR. The value of VAR[n] is either 0 or 1.
- This standard uses the two-sided z-transform as given below. See Oppenheim, A. V. and Schafer, R. W., *Digital Signal Processing*, pp. 45 - 86.

$$F(z) = \sum_{i=-\infty}^{\infty} x_i z^{-i}$$

REFERENCES

The following standards contain provisions which, through reference in this text, constitute provisions of this Standard. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this Standard are encouraged to investigate the possibility of applying the most recent editions of the standards indicated below. ANSI and TIA maintain registers of currently valid national standards published by them.

—*American National Standards:*

1. ANSI/EIA/TIA-579, Acoustic-to-Digital and Digital-to-Acoustic Transmission Requirements for ISDN Terminals, March 1991.

—*Other Standards:*

2. CCITT Recommendation G.711, *Pulse Code Modulation (PCM) of Voice Frequencies*, Vol. III, Geneva 1972.
3. CCITT Recommendation G.714, *Separate Performance Characteristics for the Encoding and Decoding Sides of PCM Channels Applicable to 4-Wire Voice-Frequency Interfaces*, Blue Book, Vol. III, Melbourne 1988.
4. IEEE Standard 269-1992, *IEEE Standard Methods for Measuring Transmission Performance of Analog and Digital Telephone Sets*, 1992.
5. IEEE Standard 661-1979, *Method for Determining Objective Loudness Ratings of Telephone Connections*, 1979.
6. ANSI J-STD-008, *Personal Station-Base Station Compatibility Requirements for 1.8 to 2.0 GHz Code Division Multiple Access (CDMA) Personal Communications Systems*.
7. TIA/EIA/IS-95-A, *Mobile Station-Base Station Compatibility Standard for Dual-Mode Wideband Spread Spectrum Cellular System*. All references to TIA/EIA/IS-95-A shall be inclusive of text adopted ty TSB74.
8. TIA/EIA/IS-125, *Recommended Minimum Performance Standard for Digital Cellular Wideband Spread Spectrum Speech Service Option 1*, May 1995.
9. TIA/EIA/IS-736, *Recommended Minimum Performance Standard for the High Rate Speech Service Option for Wideband Spread Spectrum Communication Systems*.
10. TSB74, *Telecommunications Systems Bulletin: Support for 14.4 kbps Data Rate and PCS Interaction for Wideband Spread Spectrum Cellular Systems*, December 1995.

CONTENTS

CONTENTS

CONTENTS

CONTENTS

FIGURES

TABLES

TABLES

No text.

1 GENERAL

1.1 Terms and Numeric Information

Autocorrelation Function. A function showing the relationship of a signal with a time-shifted version of itself.

Base Station. A station in the Public Radio Telecommunications Service, other than a mobile station, used for radio communications with mobile stations.

CELP. See Code Excited Linear Predictive Coding.

Codec. The combination of an encoder and decoder in series (encoder/decoder).

Code Excited Linear Predictive Coding (CELP). A speech coding algorithm. CELP coders use codebook excitation, a long-term pitch prediction filter, and a short-term formant prediction filter.

Codebook. A set of vectors used by the speech codec. For each speech codec codebook subframe, one particular vector is chosen and used to excite the speech codec's filters. The codebook vector is chosen to minimize the weighted error between the original and synthesized speech after the pitch and formant synthesis filter coefficients have been determined.

Coder. Same as "encoder."

Decoder. Generally, a device for the translation of a signal from a digital representation into an analog format. For this standard, a device which converts speech encoded in the format specified in this standard to analog or an equivalent PCM representation.

DECSD. Decoder Seed.

Encoder. Generally, a device for the translation of a signal into a digital representation. For this standard, a device which converts speech from an analog or its equivalent PCM representation to the digital representation described in this standard.

Formant. A resonant frequency of the human vocal tract causing a peak in the short term spectrum of speech.

IIR Filter. An infinite-duration impulse response filter is a filter for which the output, in response to an impulse input, never totally converges to zero. This term is usually used in reference to digital filters.

Linear Predictive Coding (LPC). A method of predicting future samples of a sequence by a linear combination of the previous samples of the same sequence. Linear Predictive Coding is frequently used in reference to a class of speech codecs.

Line Spectral Pair (LSP). A representation of digital filter coefficients in a pseudo-frequency domain. This representation has good quantization and interpolation properties.

LPC. See Linear Predictive Coding.

LSB. Least significant bit.

LSP. See Line Spectral Pair.

MSB. Most significant bit.

Mobile Station. A station in the Public Radio Telecommunications Service intended to be used while in motion or during halts at unspecified points.

Normalized Autocorrelation Function (NACF). A measure used to determine the pitch period and the degree of periodicity of the input speech. This measure is useful in distinguishing voiced from unvoiced speech.

Packet. The unit of information exchanged between service option applications in the base station and the mobile station.

Pitch. The fundamental frequency in speech caused by the periodic vibration of the human vocal cords.

RDA. Rate Determination Algorithm.

Receive Objective Loudness Rating (ROLR). A measure of receive audio sensitivity. ROLR is a frequency-weighted ratio of the line voltage input signal to a reference encoder to the acoustic output of the receiver. IEEE 269 defines the measurement of sensitivity and IEEE 661 defines the calculation of objective loudness rating.

SPL. Sound Pressure Level.

Transmit Objective Loudness Rating (TOLR). A measure of transmit audio sensitivity. TOLR is a frequency-weighted ratio of the acoustic input signal at the transmitter to the line voltage output of the reference decoder. IEEE 269 defines the measurement of sensitivity and IEEE 661 defines the calculation of objective loudness rating.

Voiced Speech. Speech generated when the vocal cords are vibrating at a fundamental frequency. Characterized by high energy, periodicity, and a large ratio of energy below 2 kHz to energy above 2 kHz.

Unvoiced Speech. Speech generated by forcing air through constrictions in the vocal tract without vibration of the vocal cords. Characterized by a lack of periodicity, and a near-unity ratio of energy below 2 kHz to energy above 2 kHz.

WAEPL. Weighted Acoustic Echo Path Loss. A measure of the echo performance under normal conversation. ANSI/EIA/TIA-579 defines the measurement of WAEPL.

Zero Input Response (ZIR). The filter output caused by the non-zero initial state of the filter when no input is present.

Zero State Response (ZSR). The filter output caused by an input when the initial state of the filter is zero.

ZIR. See Zero Input Response.

ZSR. See Zero State Response.

2 SERVICE OPTION 17: VARIABLE DATA RATE TWO-WAY VOICE

2.1 General Description

Service Option 17 provides two-way voice communications between the base station and the mobile station using the dynamically variable data rate speech codec algorithm described in this standard. The service option takes voice samples and generates an encoded speech packet for every Traffic Channel frame.[1] The receiving station generates a speech packet from every Traffic Channel frame and supplies it to the service option for decoding into voice samples.

The two speech codecs communicate at one of four rates: Rate 1, Rate 1/2, Rate 1/4, and Rate 1/8.

In case of a discrepancy between the master C simulation and the algorithmic description, the master C simulation will prevail. The master C simulation is contained in the database of the performance specification for this algorithm, TIA/EIA/IS-736.

2.2 Service Option Number

The variable data rate two-way voice service option using the speech codec algorithm described by this standard shall use service option number 17 and is called Service Option 17.

2.3 Multiplex Option

2.3.1 Required Multiplex Option Support

Service Option 17 shall support an interface with Multiplex Option 2 (see TIA/EIA/IS-95). Speech packets for Service Option 17 shall only be transported as primary traffic.

2.3.2 Interface to Multiplex Option 2

2.3.2.1 Transmitted Packets

The service option shall generate and supply exactly one packet to the multiplex sublayer every 20 ms. The packet contains the service option information bits which are transmitted as primary traffic.

The service option shall operate in one of two modes:

---

[1] IS-95 "Mobile Station-Base Station Compatibility Standard for Dual-Mode Wideband Spread Spectrum Cellular System" and J-STD-008 "Personal Station-Base Station Compatibility Requirements for 1.8 to 2.0 GHz Code Division Multiple Access (CDMA) Personal Communications Systems" use the term frame to represent a 20 ms grouping of data on the Traffic Channel. Common speech codec terminology also uses the term frame to represent a quantum of processing. For Service Option 17~~0x8000~~, the speech codec frame corresponds to speech sampled over 20 ms. The speech samples are processed into a packet. This packet is transmitted in a Traffic Channel frame.

In the first mode, the packet supplied by the service option shall be one of the 5 types shown in Table 2.3.2.1-1. Upon command, the service option shall generate Blank packets. Also, upon command, the service option shall generate a non-blank packet with a maximum rate of Rate 1/2.

In the second mode, the packet supplied by the service option shall be one of the types shown in Table 2.3.2.1-1, excluding the Rate 1 packet. Upon command, the service option shall generate a Blank packet. Also upon command, the service option shall generate a non-blank packet with a maximum rate of Rate 1/4.

Table 2.3.2.1-1. Packet Types Supplied by Service Option 17 to the Multiplex Sublayer

Packet Type	Bits per Packet
Rate 1	266
Rate 1/2	124
Rate 1/4	54
Rate 1/8	20
Blank	0

2.3.2.2 Received Packets

The multiplex sublayer in the mobile station categorizes every received Traffic Channel frame and supplies the packet type and accompanying bits, if any, to the service option as shown in Table 2.3.2.2-1. The service option processes the bits of the packet as described in 2.4. The first five received packet types shown in Table 2.3.2.2-1 correspond to the transmitted packet types shown in Table 2.3.2.1-1. When the multiplex sublayer determines that a received frame is in error, the multiplex sublayer supplies an insufficient frame quality (erasure) packet to the service option.

Table 2.3.2.2-1. Packet Types Supplied by the Multiplex Sublayer to Service Option 17

Packet Type	Bits per Packet
Rate 1	266
Rate 1/2	124
Rate 1/4	54
Rate 1/8	20
Blank	0
Insufficient frame quality (erasure)	0

2.3.3 Service Negotiation

The mobile station and base station shall perform service negotiation for the service option as described in IS-95 or J-STD-008, and the negotiated service configuration shall include only valid attributes for the service option as specified in Table 2.3.3-1.

Table 2.3.3-1. Valid Service Configuration Attributes for Service Option 17

Service Configuration Attribute	Valid Selections
Forward Multiplex Option	Multiplex Option 2
Reverse Multiplex Option	Multiplex Option 2
Forward Transmission Rates	Rate Set 2 with all four rates enabled
Reverse Transmission Rates	Rate Set 2 with all four rates enabled
Forward Traffic Type	Primary Traffic
Reverse Traffic Type	Primary Traffic

2.3.4 Initialization and Connection

2.3.4.1 Mobile Station Requirements

If the mobile station accepts a service configuration, as specified in a *Service Connect Message*, that includes a service option connection using the service option, the mobile station shall perform the following:

- If the service option connection is new (that is, not part of the previous service configuration), the mobile station shall perform speech codec initialization (see 2.4.9) at the action time associated with the *Service Connect Message*. The mobile station shall complete the initialization within 40 ms.

- Commencing at the action time associated with the *Service Connect Message*, and continuing for as long as the service configuration includes the service option connection, the service option shall process received packets and shall generate and supply packets for transmission as follows:

  - If the mobile station is in the *Conversation Substate*, the service option shall process the received packets and generate and supply packets for transmission in accordance with this standard.

  - If the mobile station is not in the *Conversation Substate*, the service option shall process the received packets in accordance with this standard, and shall generate and supply All Ones Rate 1/8 Packets for transmission, except when commanded to generate a blank packet.

2.3.4.2 Base Station Requirements

If the base station establishes a service configuration, as specified in a *Service Connect Message*, that includes a service option connection using the service option, the base station shall perform the following:

- If the service option connection is new (that is, not part of the previous service configuration), the base station shall perform speech codec initialization (see 2.4.9) no later than the action time associated with the *Service Connect Message*.
- Commencing at the action time associated with the *Service Connect Message* and continuing for as long as the service configuration includes the service option connection, the service option shall process received packets and shall generate and supply packets for transmission in accordance with this standard. The base station may defer enabling the audio input and output.

2.3.5 Service Option Control Messages

2.3.5.1 Mobile Station Requirements

The mobile station shall support one pending *Service Option Control Message* for the service option.

If the mobile station receives a *Service Option Control Message* for the service option, then, at the action time associated with the message, the mobile station shall process the message as follows:

1. If the MOBILE_TO_MOBILE field is equal to ‘1’, the service option shall process each received Blank packet as an insufficient frame quality (erasure) packet. In addition, if the INIT_CODEC field is equal to ‘1’, the service option should disable the audio output for 1 second after initialization.

   If the MOBILE_TO_MOBILE field is equal to ‘0’, the service option shall process each received packet as described in 2.4.8.

2. If the INIT_CODEC field is equal to ‘1’, the mobile station shall perform speech codec initialization (see 2.4.9). The mobile station shall complete the initialization within 40 ms.

3. If the RATE_REDUC field is equal to a value defined in Table 2.3.5.2-2, the service option shall generate the fraction of those packets normally generated as Rate 1 packets (see 2.4.4.1) at either Rate 1, Rate 1/2, or Rate 1/4 as specified by the corresponding line in Table 2.3.5.2-2. The service option shall continue to use these fractions until either of the following events occur:

   - The mobile station receives a *Service Option Control Message* specifying a different RATE_REDUC, or
   - The service option is initialized.

   The service option may use the procedure defined in 2.4.4.3 to perform this rate reduction. This rate reduction mechanism is not deterministic, but depends upon the statistics of the input speech. The values in Table 2.3.5.2-2 are based upon the assumption that 30% of active speech is unvoiced. In reduced rate level 1, unvoiced speech is encoded using Rate 1/2. In reduced rate levels 2 and 3, unvoiced speech is encoded using Rate 1/4. In reduced rate level 3, 30% of the voiced speech frames are encoded using Rate 1/2. The decision to encode the input voiced speech frame as Rate 1/2 or Rate 1 is made based upon the statistics of the input speech and the average encoding rate for active speech as defined in 2.4.4.3.

If the RATE_REDUC field is not equal to a value defined in Table 2.3.5.2-2, the mobile station shall reject the message by sending a *Mobile Station Reject Order* with the ORDQ field set equal to '00000100'.

2.3.5.2 Base Station Requirements

The base station may send a *Service Option Control Message* to the mobile station. If the base station sends a *Service Option Control Message*, the base station shall include the following type-specific fields for the service option:

Table 2.3.5.2-1. Service Option Control Message Type-Specific Fields

Field	Length (bits)
RATE_REDUC	3
RESERVED	3
MOBILE_TO_MOBILE	1
INIT_CODEC	1

RATE_REDUC - Rate reduction.

The base station shall set this field to the RATE_REDUC value from Table 2.3.5.2-2 corresponding to the rate reduction that the mobile station is to perform.

RESERVED - Reserved bits.

The base station shall set this field to '000'.

MOBILE_TO_MOBILE - Mobile-to-mobile processing.

If the mobile station is to perform mobile-to-mobile processing (see 2.3.5.1), the base station shall set this field to '1'. In addition, if the mobile station is to disable the audio output of the speech codec for 1 second after initialization, the base station shall set the INIT_CODEC field and the MOBILE_TO_-MOBILE field to '1'. If the mobile station is not to perform mobile-to-mobile processing, the base station shall set this field to '0'.

INIT_CODEC - Initialize speech codec.

If the mobile station is to initialize the speech codec (see 2.4.9), the base station shall set this field to '1'; otherwise, the base station shall set this field to '0'.

Table 2.3.5.2-2. Fraction of Packets at Rate 1, Rate 1/2, and Rate 1/4 with Rate Reduction

RATE_REDUC	Reduced Rate Mode Level	Average Encoding Rate for Active Speech (kbps)	Fraction of Normally Rate 1 Packets to be Rate 1	Fraction of Normally Rate 1 Packets to be Rate 1/2	Fraction of Normally Rate 1 Packets to be Rate 1/4
‘000’	0	14.4	1	0	0
‘001’	1	12.2	0.7	0.3	0
‘010’	2	11.2	0.7	0	0.3
‘011’	3	9.0	0.4	0.3	0.3
‘100’	4	7.2	0	1	0
All other RATE_REDUC values are reserved. Note: Average Encoding Rate calculation uses channel rates of 14.4, 7.2, and 3.6 kbps for Rate 1, 1/2, and 1/4 respectively.					

2.4 Variable Rate Speech Coding Algorithm[2]

2.4.1 Introduction

The speech codec uses a code excited linear predictive (CELP) coding algorithm. This technique uses a codebook to vector quantize the residual signal using an analysis-by-synthesis method. The speech codec produces a variable output data rate based upon speech activity. For typical two-way telephone conversations, the average data rate is reduced by a factor of two or more with respect to the maximum data rate.

The overall speech synthesis or decoder model is shown in Figure 2.4.1-1. First, a vector is taken from one of two sources depending on the rate. For Rate 1/4 and Rate 1/8 a pseudorandom vector is generated. For all other rates, a vector specified by an index $\hat{I}$ is taken from the codebook, which is a table of vectors. This vector is multiplied by a gain term $\hat{G}$, and then is filtered by the long-term pitch synthesis filter whose characteristics are governed by the pitch parameters $\hat{L}$ and $\hat{b}$. The output of the pitch synthesis filter is processed by the pitch pre-filter. The pitch pre-filter parameters are the pitch lag, $\hat{L}$, and an attenuated pitch gain coefficient, $\hat{b}'$, derived from $\hat{b}$. The output of the pre-filter is

[2]For a summary of Service Option 17~~0x8000~~ notation, see 2.5.

filtered by the formant synthesis filter[3] to reproduce the speech signal. The output of the formant synthesis filter is filtered by the adaptive postfilter, PF(z).

The speech codec encoding procedure involves determining the input parameters for the decoder which minimize the perceptual difference between the synthesized and the original speech. The selection processes for each set of parameters are described in this section. The encoding procedure also includes quantizing the parameters and packing them into data packets for transmission.

The speech codec decoding procedure involves unpacking the data packets, unquantizing the received parameters, and reconstructing the speech signal from these parameters. The reconstruction consists of filtering the scaled codebook vector, $c_d(n)$, as shown in Figure 2.4.1-1.

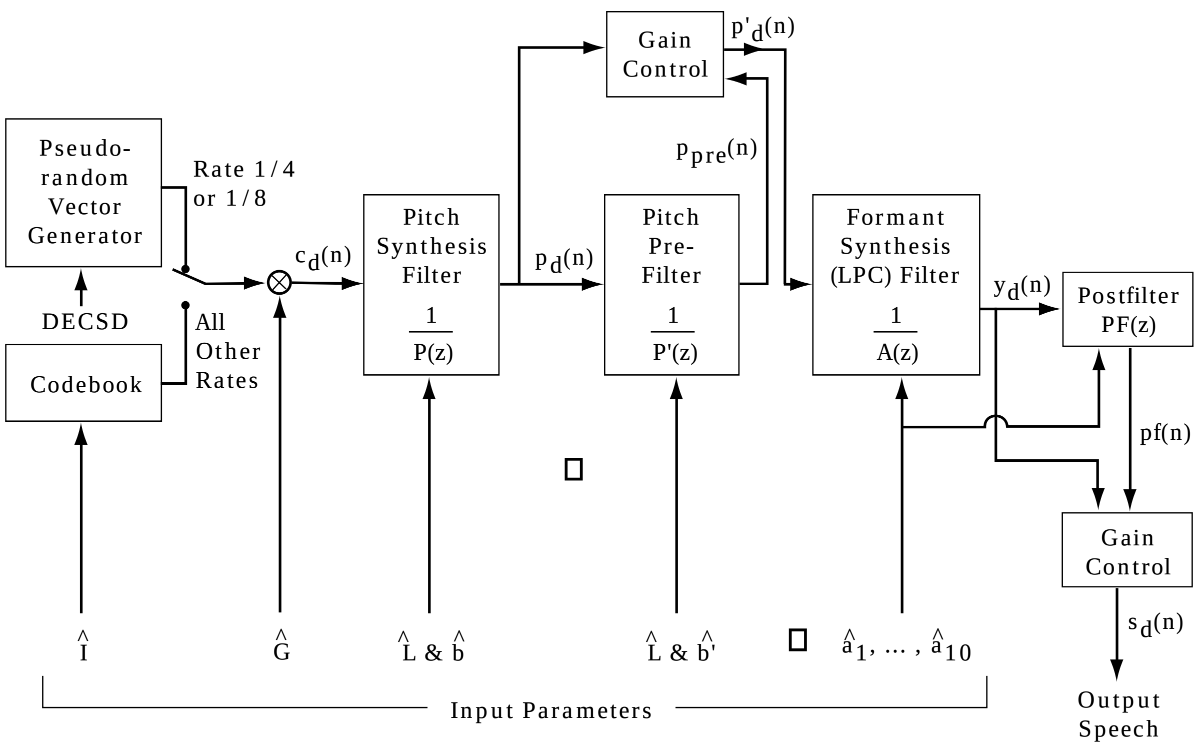


Figure 2.4.1-1. Speech Synthesis Structure in the Receiving Speech Codec

The input speech is sampled at 8 kHz. This speech is broken down into 20 ms speech codec frames, each consisting of 160 samples. The formant synthesis (LPC) filter coefficients are updated once per frame, regardless of the data rate selected. The number of bits used to encode the LPC parameters is a function of the selected data rate. Within each

---

[3]Also called the linear predictive coding filter, whose characteristics are governed by the filter coefficients $\hat{a}_1, \ldots, \hat{a}_{10}$.

frame, the pitch and codebook parameters are updated a varying number of times, depending upon the selected data rate. Table 2.4.1-1 describes the various parameters used for each rate.

Table 2.4.1-1. Parameters Used for Each Rate

Parameter	Rate 1	Rate 1/2	Rate 1/4	Rate 1/8
Linear predictive coding (LPC) updates per frame	1	1	1	1
Samples per LPC update, L_A	160 (20 ms)	160 (20 ms)	160 (20 ms)	160 (20 ms)
Bits per LPC update	32	32	32	10
Pitch updates (subframes) per frame	4	4	0	0
Samples per pitch subframe, L_p	40 (5 ms)	40 (5 ms)	-	–
Bits per pitch update	11	11	-	–
Codebook updates (subframes) per frame	16	4	5	1
Samples per codebook subframe, L_C	10 (1.25 ms)	40 (5 ms)	32 (4 ms)	160 (20 ms)
Bits per codebook update	11.75*	12	4*	6*

*Note:

Rate 1 uses 12 bits per codebook update in 12 of the 16 codebook subframes per frame and 11 bits per codebook update, in four codebook subframes.

Rate 1/4 uses five unsigned codebook gains, each 4-bits long for scaling the pseudorandom excitation.

Rate 1/8 uses six bits for pseudorandom excitation, instead of using the codebook.

The components for each rate packet are shown in Figures 2.4.1-2 through 2.4.1-5. In these figures, each LPC frame corresponds to one 160-sample frame of speech.

The number in the LPC block of each figure is the number of bits used at that rate to encode the LPC coefficients. Each pitch block corresponds to a pitch update within each frame, and the number in each pitch block corresponds to the number of bits used to encode the updated pitch parameters. For example at Rate 1, the pitch parameters are updated four times, once for each quarter of the speech frame, each time using 11 bits to encode the new pitch parameters. Similarly, each codebook block corresponds to a codebook update within each frame, and the number in each codebook block corresponds to the number of bits used to encode the updated codebook parameters. For example at Rate 1/2, the codebook parameters are updated four times, once for each quarter of the speech frame, each time using 12 bits to encode the parameters.

LPC Frame	32																Total = 264 bits
Pitch Subframe	11				11				11				11				+
Codebook Subframe	12	12	12	11	12	12	12	11	12	12	12	11	12	12	12	11	2 reserved bits

Figure 2.4.1-2. Bit Allocation for a Rate 1 Packet

LPC Frame	32				
Pitch Subframe	11	11	11	11	Total = 124 bits
Codebook Subframe	12	12	12	12	

Figure 2.4.1-3. Bit Allocation for a Rate 1/2 Packet

LPC Frame	32					Total = 52 bits
Pitch Subframe	0					+
Codebook Subframe	4	4	4	4	4	2 reserved bits

Figure 2.4.1-4. Bit Allocation for a Rate 1/4 Packet

LPC Frame	10	Total = 16 bits
Pitch Subframe	0	+
Codebook Subframe	6	4 reserved bits

Figure 2.4.1-5. Bit Allocation for a Rate 1/8 Packet

Table 2.4.1-2 lists all the parameter codes transmitted for each rate packet. The following list describes each parameter:

LSPi	Line Spectral Pair frequency i.
LSPVi	Line Spectral Pair frequencies grouped into five vectors of dimension two.
PLAGi	Pitch Lag for the ith pitch subframe.
PFRACi	Fractional Pitch Lag for the ith pitch subframe.
PGAINi	Pitch Gain for the ith pitch subframe.
CBINDEXi	Codebook Index for the ith codebook subframe.
CBGAINi	Unsigned Codebook Gain for the ith codebook subframe.
CBSEED	Random Seed for Rate 1/8 packets.
CBSIGNi	Sign of the Codebook Gain for the ith codebook subframe.

This standard refers to the LSB of a particular code as CODE[0] and the more significant bits as CODE[1], CODE[2], etc. For example, if LSPV1 = ‘001011’ in binary for a maximum rate frame, LSPV1[0] = ‘1’, LSPV1[1] = ‘1’, LSPV1[2] = ‘0’, LSPV1[3] = ‘1’, LSPV1[4] = ‘0’, and LSPV1[5] = ‘0’.

Table 2.4.1-2. Transmission Codes and Bit Allocations (Part 1 of 2)

	Rate					Rate			
Code	**1**	**1/2**	**1/4**	**1/8**	**Code**	**1**	**1/2**	**1/4**	**1/8**
LSP1	—	—	—	1	CBINDEX3	7	7	—	—
LSP2	—	—	—	1	CBINDEX4	7	7	—	—
LSP3	—	—	—	1	CBINDEX5	7	—	—	—
LSP4	—	—	—	1	CBINDEX6	7	—	—	—
LSP5	—	—	—	1	CBINDEX7	7	—	—	—
LSP6	—	—	—	1	CBINDEX8	7	—	—	—
LSP7	—	—	—	1	CBINDEX9	7	—	—	—
LSP8	—	—	—	1	CBINDEX10	7	—	—	—
LSP9	—	—	—	1	CBINDEX11	7	—	—	—
LSP10	—	—	—	1	CBINDEX12	7	—	—	—
LSPV1	6	6	6	—	CBINDEX13	7	—	—	—
LSPV2	7	7	7	—	CBINDEX14	7	—	—	—
LSPV3	7	7	7	—	CBINDEX15	7	—	—	—
LSPV4	6	6	6	—	CBINDEX16	7	—	—	—
LSPV5	6	6	6	—	CBGAIN1	4	4	4	2
PLAG1	7	7	—	—	CBGAIN2	4	4	4	—
PLAG2	7	7	—	—	CBGAIN3	4	4	4	—
PLAG3	7	7	—	—	CBGAIN4	3	4	4	—
PLAG4	7	7	—	—	CBGAIN5	4	—	4	—
PFRAC1	1	1	—	—	CBGAIN6	4	—	—	—
PFRAC2	1	1	—	—	CBGAIN7	4	—	—	—
PFRAC3	1	1	—	—	CBGAIN8	3	—	—	—
PFRAC4	1	1	—	—	CBGAIN9	4	—	—	—
PGAIN1	3	3	—	—	CBGAIN10	4	—	—	—
PGAIN2	3	3	—	—	CBGAIN11	4	—	—	—
PGAIN3	3	3	—	—	CBGAIN12	3	—	—	—
PGAIN4	3	3	—	—	CBGAIN13	4	—	—	—
CBSEED	—	—	—	4	CBGAIN14	4	—	—	—
CBINDEX1	7	7	—	—	CBGAIN15	4	—	—	—
CBINDEX2	7	7	—	—	CBGAIN16	3	—	—	—

Table 2.4.1-2. Transmission Codes and Bit Allocations (Part 2 of 2)

	Rate					Rate			
Code	**1**	**1/2**	**1/4**	**1/8**	**Code**	**1**	**1/2**	**1/4**	**1/8**
CBSIGN1	1	1	—	—	CBSIGN9	1	—	—	—
CBSIGN2	1	1	—	—	CBSIGN10	1	—	—	—
CBSIGN3	1	1	—	—	CBSIGN11	1	—	—	—
CBSIGN4	1	1	—	—	CBSIGN12	1	—	—	—
CBSIGN5	1	—	—	—	CBSIGN13	1	—	—	—
CBSIGN6	1	—	—	—	CBSIGN14	1	—	—	—
CBSIGN7	1	—	—	—	CBSIGN15	1	—	—	—
CBSIGN8	1	—	—	—	CBSIGN16	1	—	—	—

2.4.2 Input Audio Interface

2.4.2.1 Input Audio Interface in the Mobile Station

The input audio may be either an analog or digital signal.

2.4.2.1.1 Conversion and Scaling

The speech shall be sampled at a rate of 8000 samples per second. The speech shall be quantized to a uniform PCM format with at least 13 magnitude bits of dynamic range.

The quantities in this standard assume a 14-bit integer input quantization with a range of ±8031. The following speech codec discussion assumes this 14-bit integer quantization. If the speech codec uses a different quantization, then appropriate scaling should be used.

2.4.2.1.2 Digital Audio Input

If the input audio is an 8-bit μlaw PCM signal, it shall be converted to a uniform PCM format according to Table 2 in CCITT Recommendation G.711 "Pulse Code Modulation (PCM) of Voice Frequencies."

2.4.2.1.3 Analog Audio Input

If the input is in analog form, the mobile station shall sample the analog speech and shall convert the samples to a digital format for speech codec processing. This shall be done by either the following or an equivalent method: First, the input gain audio level is adjusted. Then, the signal is bandpass filtered to prevent aliasing. Finally, the filtered signal is sampled and quantized (see 2.4.2.1.1).

2.4.2.1.3.1 Adjusting the Transmit Level

The mobile station shall have a transmit objective loudness rating (TOLR) equal to -46 dB, when transmitting to a reference base station (see 2.4.10.2.1). The loudness ratings are described in IEEE Standard 661-1979 "IEEE Standard Method for Determining Objective Loudness Ratings of Telephone Connections." Measurement techniques and tolerances are described in IS-125 "Recommended Minimum Performance Standard for Wideband Spread Spectrum Digital Cellular System Speech Service Options."

2.4.2.1.3.2 Band Pass Filtering

Input anti-aliasing filtering shall conform to CCITT Recommendation G.714 "Separate Performance Characteristics for the Encoding and Decoding Sides of PCM Channels Applicable to 4-Wire Voice-Frequency Interfaces." Additional anti-aliasing filtering may be provided by the manufacturer.

2.4.2.1.3.3 Echo Return Loss

Provision shall be made to ensure adequate isolation between receive and transmit audio paths in all modes of operation. When no external transmit audio is present, the speech codec shall not generate packets at rates higher than Rate 1/8 (see 2.4.4), due to acoustic coupling of the receive audio into the transmit audio path (specifically with the receive audio at full volume). Target levels of 45 dB WAEPL should be met. See ANSI/EIA/TIA Standard 579 "Acoustic-to-Digital and Digital-to-Acoustic Transmission Requirements for ISDN Terminals." Refer to the requirements stated in IS-125 "Recommended Minimum Performance Standard for Wideband Spread Spectrum Digital Cellular System Speech Service Options."

2.4.2.2 Input Audio Interface in the Base Station

2.4.2.2.1 Sampling and Format Conversion

The base station converts the input speech (analog, μlaw companded Pulse Code Modulation, or other format) into a uniform quantized PCM format with at least 13 magnitude bits of dynamic range. The sampling rate is 8000 samples per second. The sampling and conversion process shall be as in 2.4.2.1.1.

2.4.2.2.2 Adjusting the Transmit Level

The base station shall set the transmit level so that a 1004 Hz tone at a level of 0 dBm0 at the network interface produces a level 3.17 dB below the level of a sine wave whose peak is at the maximum quantization level. Measurement techniques and tolerances are described in IS-125 "Recommended Minimum Performance Standard for Wideband Spread Spectrum Digital Cellular System Speech Service Options."

2.4.2.2.3 Echo Canceling

The base station shall provide a method to cancel echoes returned by the PSTN interface.[4] The echo canceling function should provide at least 30 dB of echo return loss enhancement. The echo canceling function should work over a range of PSTN echo return delays from 0 to 48 ms.

---

[4]Because of the relatively long delays inherent in the speech coding and transmitting processes, echoes that are not sufficiently suppressed are noticeable to the mobile station user.

2.4.2.2.4 Ear Protection

To protect the user from possible ear damage, ear-piece acoustic output shall be limited so as not to exceed 120 dB SPL when placed to the ear as measured in accordance with 7.11 of IEEE 269-1992 “Standard Method for Measuring Transmission Performance on Analog and Digital Telephone Sets.”

2.4.3 Determining the Formant Prediction Parameters

2.4.3.1 Form of the Formant Synthesis Filter

The formant synthesis filter, which is similar to the traditional LPC formant synthesis filter, is the inverse of the formant prediction error filter. The prediction error filter is of the tenth order (i.e., P is equal to 10), and has transfer function

$$A(z) = 1 - \sum_{i=1}^{P} a_i z^{-i} \tag{2.4.3.1-1}$$

The formant synthesis filter has transfer function

$$\frac{1}{A(z)} = \frac{1}{1 - \sum_{i=1}^{P} a_i z^{-i}} \tag{2.4.3.1-2}$$

The LPC coefficients, a_i, are computed from the input speech.

2.4.3.2 Encoding

The encoding process begins by determining the formant prediction parameters. This is performed by the following steps:

1. High-pass filter the input samples.
2. Window the filtered samples using a Hamming window.
3. Compute the 17 values of the autocorrelation function corresponding to shifts from 0 to 16 samples.
4. Determine the LPC coefficients from the autocorrelation values.
5. Transform the LPC coefficients to LSP frequencies.
6. Convert the LSP frequencies into LSP codes (these codes are placed into the packet for transmission).

2.4.3.2.1 High-Pass Filtering of Input Samples

A high-pass digital filter is inserted into the input signal path to remove unwanted background and circuit noise and to prevent a DC offset from artificially increasing R(0) (see 2.4.3.2.3) and thus disrupting the rate decision algorithm (see 2.4.4). One possible high-pass filter for accomplishing these objectives is defined as

$$HPF(z) = 0.94615 \frac{z^2 - 2z + 1}{z^2 - 1.88z + 0.8836} \tag{2.4.3.2.1-1}$$

2.4.3.2.2 Windowing the Samples

The high-pass filtered speech samples are windowed using a Hamming window which is centered at the center of the fourth Rate 1 pitch subframe. The window is 160 samples long (i.e., L_A is equal to 160).

Let s(n) be the input speech signal with the DC removed, where s(0) denotes the first sample of the current frame. The windowed speech signal is defined as

$$S_w(n) = s(n + 60)W_H(n), \qquad 0 \le n \le L_A - 1 \tag{2.4.3.2.2-1}$$

where the Hamming window, $W_H(n)$, is defined in Table 2.4.3.2.2-1 in hexadecimal format. Each value in the table has 14 fractional bits.

Note the offset of 60 samples, which results in the window of speech being centered between the 139th and 140th samples of the current speech frame of 160 samples, and s(160+i) for $0 \le i \le 59$ are the first 60 samples of the next speech frame.

Table 2.4.3.2.2-1. Hamming Window Values $W_H(n)$

n	$W_H(n)$	n
0	0x051f	159
1	0x0525	158
2	0x0536	157
3	0x0554	156
4	0x057d	155
5	0x05b1	154
6	0x05f2	153
7	0x063d	152
8	0x0694	151
9	0x06f6	150
10	0x0764	149
11	0x07dc	148
12	0x085e	147
13	0x08ec	146
14	0x0983	145
15	0x0a24	144
16	0x0ad0	143
17	0x0b84	142
18	0x0c42	141
19	0x0d09	140
20	0x0dd9	139
21	0x0eb0	138
22	0x0f90	137
23	0x1077	136
24	0x1166	135
25	0x125b	134
26	0x1357	133
27	0x1459	132
28	0x1560	131
29	0x166d	130
30	0x177f	129
31	0x1895	128
32	0x19af	127
33	0x1acd	126
34	0x1bee	125
35	0x1d11	124
36	0x1e37	123
37	0x1f5e	122
38	0x2087	121
39	0x21b0	120
40	0x22da	119
41	0x2403	118
42	0x252d	117
43	0x2655	116
44	0x277b	115
45	0x28a0	114
46	0x29c2	113
47	0x2ae1	112
48	0x2bfd	111
49	0x2d15	110
50	0x2e29	109
51	0x2f39	108
52	0x3043	107
53	0x3148	106
54	0x3247	105
55	0x333f	104
56	0x3431	103
57	0x351c	102
58	0x3600	101
59	0x36db	100
60	0x37af	99
61	0x387a	98
62	0x393d	97
63	0x39f6	96
64	0x3aa6	95
65	0x3b4c	94
66	0x3be9	93
67	0x3c7b	92
68	0x3d03	91
69	0x3d80	90
70	0x3df3	89
71	0x3e5b	88
72	0x3eb7	87
73	0x3f09	86
74	0x3f4f	85
75	0x3f89	84
76	0x3fb8	83
77	0x3fdb	82
78	0x3ff3	81
79	0x3fff	80

2.4.3.2.3 Computing the Autocorrelation Function

Following the windowing operation, the kth value of the autocorrelation function is computed as

$$R(k) = \sum_{m=0}^{L_A-1-k} S_w(m) S_w(m+k), \qquad 0 \le k \le 16 \tag{2.4.3.2.3-1}$$

Only the first 17 values of the autocorrelation function, R(0) through R(16), need to be computed from the windowed speech signal within the analysis window. Of these, the first 11 values of the autocorrelation function are required for LPC analysis. All 17 values are used for the rate determination algorithm defined in 2.4.4.1.

2.4.3.2.4 Determining the LPC Coefficients from the Autocorrelation Function

The LPC coefficients are obtained from the autocorrelation function. A method is Durbin's recursion, as shown below.[5]

```
{
  E^(0) = R(0)
  i = 1
  while (i ≤ P)
    {
      k_i = { R(i) - Σ_{j=1}^{i-1} α_j^(i-1) R(i - j) } /E^(i - 1)
      α_i^(i) = k_i
      j = 1
      while (j ≤ i-1)
        {
          α_j^(i) = α_j^(i-1) - k_i α_{i-j}^(i-1)
          j = j + 1
        }
      E^(i) = (1 - k_i^2) E^(i - 1)
      i = i + 1
    }
}
```

The LPC coefficients are

$$a_j = \alpha^{(P)}_j, \qquad 1 \le j \le P \tag{2.4.3.2.4-1}$$

---

[5] See Rabiner, L. R. and Schafer, R. W., *Digital Processing of Speech Signals,* (New Jersey: Prentice-Hall Inc, 1978), pp. 411-412. The superscripts in parentheses represent the stage of Durbin's recursion. For example $\alpha_j^{(i)}$ refers to α_j at the ith stage.

2.4.3.2.5 Transforming the LPC Coefficients to Line Spectrum Pairs (LSPs)

The LPC coefficients are transformed into line spectrum pair frequencies.

The prediction error filter transfer function, A(z), is given by

$$A(z) = 1 - a_1 z^{-1} - \ldots - a_{10} z^{-10} \quad (2.4.3.2.5\text{-}1)$$

where a_i, $1 \le i \le 10$, are the LPC coefficients as described earlier.

Define two new transfer functions $P_A(z)$ and $Q_A(z)$ as

$$P_A(z) = A(z) + z^{-11} A\left(z^{-1}\right) = 1 + p_1 z^{-1} + \ldots + p_5 z^{-5} + p_5 z^{-6} + \ldots + p_1 z^{-10} + z^{-11} \quad (2.4.3.2.5\text{-}2)$$

and

$$Q_A(z) = A(z) - z^{-11} A\left(z^{-1}\right) = 1 + q_1 z^{-1} + \ldots + q_5 z^{-5} - q_5 z^{-6} - \ldots - q_1 z^{-10} - z^{-11} \quad (2.4.3.2.5\text{-}3)$$

where

$$p_i = -a_i - a_{11-i}, \qquad 1 \le i \le 5 \quad (2.4.3.2.5\text{-}4)$$

and

$$q_i = -a_i + a_{11-i}, \qquad 1 \le i \le 5 \quad (2.4.3.2.5\text{-}5)$$

The LSP frequencies are the ten roots which exist between w=0 and w=1.0 in the following two equations:

$$P'(w) = \cos(5(\pi w)) + p'_1 \cos(4(\pi w)) + \ldots + p'_4 \cos(\pi w) + \frac{p'_5}{2} \quad (2.4.3.2.5\text{-}6)$$

$$Q'(w) = \cos(5(\pi w)) + q'_1 \cos(4(\pi w)) + \ldots + q'_4 \cos(\pi w) + \frac{q'_5}{2} \quad (2.4.3.2.5\text{-}7)$$

where the parameters p' and q' are computed recursively from the parameters p and q as

$$p'_0 = q'_0 = 1 \quad (2.4.3.2.5\text{-}8)$$

$$p'_i = p_i - p'_{i-1}, \qquad 1 \le i \le 5 \quad (2.4.3.2.5\text{-}9)$$

$$q'_i = q_i + q'_{i-1}, \qquad 1 \le i \le 5 \quad (2.4.3.2.5\text{-}10)$$

Since the formant synthesis (LPC) filter is stable, the roots of the two functions alternate in the range from 0 to 1.0. If these ten roots are denoted as $w_1, w_2, \ldots, w_{10}$ in the increasing order of magnitude, then w_i for i=1,3,5,7,9 are roots of P'(w) and w_i for i=2,4,6,8,10 are those of Q'(w).

2.4.3.2.6 Converting the LSP Frequencies to Transmission Codes for Rate 1, Rate 1/2, and Rate 1/4

For Rate 1, Rate 1/2, and Rate 1/4, a vector quantizer (VQ) is used to quantize the 10 LSP frequencies into 32 bits. The quantization procedure is described in the following subsections.

2.4.3.2.6.1 Computing the Sensitivities of the LSP Frequencies

Before quantization begins, the following algorithm is used to compute how sensitive each LSP is to quantization. These “sensitivity weightings” are used in the quantization process to weight the quantization error in each LSP frequency appropriately:

First, obtain the set of values J_i, composed of $J_i(1)$ through $J_i(10)$, where i is the index of the LSP frequency of interest, by performing long division operations on $P_A(z)$ and $Q_A(z)$ given in Equations 2.4.3.2.5-2 and 2.4.3.2.5-3. For the LSP frequencies with odd index, w_1, w_3, etc., the long division is performed as

$$\frac{1 + p_1 z^{-1} + p_2 z^{-2} + \ldots. + p_1 z^{-10} + z^{-11}}{1 - 2\cos(\pi w_i) z^{-1} + z^{-2}} = J_i(1) + J_i(2) z^{-1} + \ldots + J_i(10) z^{-9} \qquad (2.4.3.2.6.1\text{-}1)$$

and for the LSP frequencies with even index, w_2, w_4, etc., the long division is performed as

$$\frac{1 + q_1 z^{-1} + q_2 z^{-2} + \ldots. - q_1 z^{-10} - z^{-11}}{1 - 2\cos(\pi w_i) z^{-1} + z^{-2}} = J_i(1) + J_i(2) z^{-1} + \ldots + J_i(10) z^{-9} \qquad (2.4.3.2.6.1\text{-}2)$$

Next, compute the autocorrelations of the vectors J_i, using the following equation:

$$R_{J_i}(n) = \sum_{k=1}^{10-n} J_i(k) J_i(k+n), \qquad 0 \le n < 10 \ \text{ and } \ 1 \le i \le 10 \qquad (2.4.3.2.6.1\text{-}3)$$

Finally, compute the sensitivity weights for the LSP frequencies by cross correlating the R_{J_i} vectors with the autocorrelation vector computed from the speech (see Equation 2.4.3.2.3-1) and multiplying the results by $\sin^2(\pi w_i)$. The final sensitivity weights, SW_i are given by

$$SW_i = \sin^2(\pi w_i)\left(R(0) R_{J_i}(0) + 2.0 \sum_{k=1}^{9} R(k) R_{J_i}(k) \right), \qquad 1 \le i \le 10 \qquad (2.4.3.2.6.1\text{-}4)$$

Use these weights, SW_i, to compute the weighted square error distortion metrics needed to search the LSP VQ codebooks, as described in the next subsection.

2.4.3.2.6.2 Vector Quantizing the LSP Frequencies

In the LSP VQ algorithm, the 10-dimensional LSP vector is partitioned into five 2-dimensional subvectors. Each of these 2-dimensional subvectors is quantized by a VQ, whose codebooks vary in size.

Define w_i as the ith LSP frequency and wq_i as the quantized ith LSP frequency. The VQ codebook values are given in tables in 2.4.3.2.6.3. Define $L_k(i,j)$ as the jth element of the kth vector in the ith VQ codebook. For example, $L_{23}(3,1)$ is the first element of the 23rd vector in codebook 3, shown in Table 2.4.3.2.6.3-3 as 0.2393.

The vectors in the vector quantizer codebooks are differential vectors; i.e., the VQ codebooks contain possible values for the quantized differences in the LSP frequencies, given by $\Delta w_i = w_i - w_{i-1}$. The five subvectors are quantized sequentially in the following manner.

The first VQ codebook contains possible quantized values for $\Delta w_1 = w_1 - w_0 = w_1$ and $\Delta w_2 = w_2 - w_1$. The best vector in the first codebook is selected as the vector which minimizes the sensitivity weighted error between the quantized and unquantized LSP frequencies in the first subvector, which is computed by

$$\begin{aligned} \text{error} &= SW_1(w_1 - wq_1)^2 + SW_2(w_2 - wq_2)^2 \\ &= SW_1\big(w_1 - (\Delta wq_1)\big)^2 + SW_2\big(w_2 - (\Delta wq_1 + \Delta wq_2)\big)^2 \\ &= SW_1\big(w_1 - (L_k(1,1))\big)^2 + SW_2\big(w_2 - (L_k(1,1) + L_k(1,2))\big)^2 \end{aligned} \tag{2.4.3.2.6.2-1}$$

This error function is computed for each of the 64 codevectors in the first LSP VQ codebook (i.e., $0 \le k < 64$). The codevector which results in the minimum error is selected, and the 6-bit LSPV1 transmission code is set equal to the index of this codevector. Define the index of the best vector for the ith codebook as kbst(i). Once kbst(1) has been determined, the first two quantized LSP frequencies can be reconstructed from the first VQ codebook as

$$\begin{aligned} wq_1 &= \Delta wq_1 = L_{kbst(1)}(1,1) \\ wq_2 &= \Delta wq_1 + \Delta wq_2 = L_{kbst(1)}(1,1) + L_{kbst(1)}(1,2) \end{aligned} \tag{2.4.3.2.6.2-2}$$

The remaining subvectors are quantized sequentially in a similar manner. The ith VQ codebook contains possible quantized values for $\Delta w_{2i-1} = w_{2i-1} - w_{2i-2}$ and $\Delta w_{2i} = w_{2i} - w_{2i-1}$. The best vector in the ith codebook is selected as the vector which minimizes the sensitivity weighted error between the quantized and unquantized LSP frequencies in the ith subvector, computed by

$$\begin{aligned} \text{error} &= SW_{2i-1}(w_{2i-1} - wq_{2i-1})^2 + SW_{2i}(w_{2i} - wq_{2i})^2 \\ &= SW_{2i-1}\big(w_{2i-1} - (wq_{2i-2} + \Delta wq_{2i-1})\big)^2 + SW_{2i}\big(w_{2i} - (wq_{2i-2} + \Delta wq_{2i-1} + \Delta wq_{2i})\big)^2 \\ &= SW_{2i-1}\big(w_{2i-1} - (wq_{2i-2} + L_k(i,1))\big)^2 + SW_{2i}\big(w_{2i} - (wq_{2i-2} + L_k(i,1) + L_k(i,2))\big)^2 \end{aligned} \tag{2.4.3.2.6.2-3}$$

This error function is computed for each of the codevectors in the ith LSP VQ codebook. The index of the codevector which results in the minimum error, kbst(i), is selected and the LSPVi transmission code is set equal to kbst(i). The two quantized LSP frequencies in the ith subvector can be reconstructed from the ith VQ codebook and the previously quantized LSP frequencies as

$$\begin{aligned} wq_{2i-1} &= wq_{2i-2} + \Delta wq_{2i-2} = wq_{2i-2} + L_{kbst(i)}(i,1) \\ wq_{2i} &= wq_{2i-2} + \Delta wq_{2i-1} + \Delta wq_{2i} = wq_{2i-2} + L_{kbst(i)}(i,1) + L_{kbst(i)}(i,2) \end{aligned} \qquad (2.4.3.2.6.2\text{-}4)$$

This algorithm is performed sequentially for each of the five subvectors, until all of the subvectors have been quantized and all five LSPVi transmission codes have been determined.

The state of the LSP predictor $P_{w_i}(z)$ (see 2.4.3.2.7) is set equal to the reconstructed LSP frequencies wq_i.

2.4.3.2.6.3 LSP VQ Codebooks

The vector quantization codebooks required for Rate 1, Rate 1/2, and Rate 1/4 encoding are given in Tables 2.4.3.2.6.3-1 through 2.4.3.2.6.3-5. Floating-point values shall be quantized to fixed-point precision using

$$X_{int} = \frac{\left(\mathrm{round}\left(2^{14} X_{float}\right)\right)}{2^{14}} \qquad (2.4.3.2.6.3\text{-}1)$$

where X_{float} is the value from Tables 2.4.3.2.6.3-1 through 2.4.3.2.6.3-5 and X_{int} is the fixed-point precision number that shall be used.

Table 2.4.3.2.6.3-1. LSP Vector Quantization Table for LSPVQ1

Index	(x,y)	Index	(x,y)	Index	(x,y)	Index	(x,y)
0	0.0327 0.0118	16	0.0471 0.0215	32	0.0386 0.0130	48	0.0415 0.0200
1	0.0919 0.0111	17	0.1046 0.0125	33	0.0962 0.0119	49	0.1018 0.0088
2	0.0427 0.0440	18	0.0645 0.0298	34	0.0542 0.0387	50	0.0681 0.0339
3	0.1327 0.0185	19	0.1599 0.0160	35	0.1431 0.0185	51	0.1436 0.0325
4	0.0469 0.0050	20	0.0593 0.0039	36	0.0526 0.0051	52	0.0555 0.0122
5	0.1272 0.0091	21	0.1187 0.0462	37	0.1175 0.0260	53	0.1042 0.0485
6	0.0892 0.0059	22	0.0749 0.0341	38	0.0831 0.0167	54	0.0826 0.0345
7	0.1771 0.0193	23	0.1520 0.0511	39	0.1728 0.0510	55	0.1374 0.0743
8	0.0222 0.0158	24	0.0290 0.0792	40	0.0273 0.0437	56	0.0383 0.1018
9	0.1100 0.0127	25	0.0909 0.0362	41	0.1172 0.0113	57	0.1005 0.0358
10	0.0827 0.0055	26	0.0753 0.0081	42	0.0771 0.0144	58	0.0704 0.0086
11	0.0978 0.0791	27	0.1111 0.1058	48	0.1122 0.0751	59	0.1301 0.0586
12	0.0665 0.0047	28	0.0519 0.0253	44	0.0619 0.0119	60	0.0597 0.0241
13	0.0700 0.1401	29	0.0828 0.0839	45	0.0492 0.1276	61	0.0832 0.0621
14	0.0670 0.0859	30	0.0685 0.0541	46	0.0658 0.0695	62	0.0555 0.0573
15	0.1913 0.1048	31	0.1421 0.1258	47	0.1882 0.0615	63	0.1504 0.0839

Table 2.4.3.2.6.3-2. LSP Vector Quantization Table for LSPVQ2 (Part 1 of 2)

Index	(x,y)	Index	(x,y)	Index	(x,y)	Index	(x,y)
0	0.0255 0.0293	22	0.0706 0.1732	44	0.0588 0.0916	66	0.0265 0.1231
1	0.0904 0.0219	23	0.2656 0.0401	45	0.1110 0.1116	67	0.1495 0.0573
2	0.0151 0.1211	24	0.0418 0.0745	46	0.0224 0.2719	68	0.0566 0.0262
3	0.1447 0.0498	25	0.0762 0.1038	47	0.1633 0.2220	69	0.1569 0.0293
4	0.0470 0.0253	26	0.0583 0.1748	48	0.0402 0.0520	70	0.1341 0.1144
5	0.1559 0.0177	27	0.1746 0.1285	49	0.1061 0.0448	71	0.2271 0.0544
6	0.1547 0.0994	28	0.0527 0.1169	50	0.0402 0.1352	72	0.0214 0.0877
7	0.2394 0.0242	29	0.1314 0.0830	51	0.1499 0.0775	73	0.0847 0.0719
8	0.0091 0.0813	30	0.0556 0.2116	52	0.0664 0.0589	74	0.0794 0.1384
9	0.0857 0.0590	31	0.1073 0.2321	53	0.1081 0.0727	75	0.2067 0.0274
10	0.0934 0.1326	32	0.0297 0.0570	54	0.0801 0.2206	76	0.0703 0.0688
11	0.1889 0.0282	33	0.0981 0.0403	55	0.2165 0.1157	77	0.1099 0.1306
12	0.0813 0.0472	34	0.0468 0.1103	56	0.0566 0.0802	78	0.0391 0.2947
13	0.1057 0.1494	35	0.1740 0.0243	57	0.0911 0.1116	79	0.2024 0.1670
14	0.0450 0.3315	36	0.0725 0.0179	58	0.0306 0.1703	80	0.0471 0.0525
15	0.2163 0.1895	37	0.1255 0.0474	59	0.1792 0.0836	81	0.1245 0.0290
16	0.0538 0.0532	38	0.1374 0.1362	60	0.0655 0.0999	82	0.0264 0.1557
17	0.1399 0.0218	39	0.1922 0.0912	61	0.1061 0.1038	83	0.1568 0.0807
18	0.0146 0.1552	40	0.0285 0.0947	62	0.0298 0.2089	84	0.0718 0.0399
19	0.1755 0.0626	41	0.0930 0.0700	63	0.1110 0.1753	85	0.1193 0.0685
20	0.0822 0.0202	42	0.0593 0.1372	64	0.0361 0.0311	86	0.0883 0.1594
21	0.1299 0.0663	43	0.1909 0.0576	65	0.0970 0.0239	87	0.2729 0.0764

Table 2.4.3.2.6.3-2. LSP Vector Quantization Table for LSPVQ2 (Part 2 of 2)

Index	(x,y)	Index	(x,y)	Index	(x,y)	Index	(x,y)
88	0.0500 0.0754	98	0.0349 0.1253	108	0.0720 0.0816	118	0.0861 0.1855
89	0.0809 0.1108	99	0.1653 0.0507	109	0.1240 0.1089	119	0.1764 0.1500
90	0.0541 0.1648	100	0.0625 0.0354	110	0.0439 0.2475	120	0.0444 0.0970
91	0.1523 0.1385	101	0.1376 0.0431	111	0.1498 0.2040	121	0.0935 0.0903
92	0.0614 0.1196	102	0.1187 0.1465	112	0.0336 0.0718	122	0.0424 0.1687
93	0.1209 0.0847	103	0.2164 0.0872	113	0.1213 0.0187	123	0.1633 0.1102
94	0.0345 0.2242	104	0.0360 0.0974	114	0.0451 0.1450	124	0.0793 0.0897
95	0.1442 0.1747	105	0.1008 0.0698	115	0.1368 0.0885	125	0.1060 0.0897
96	0.0199 0.0560	106	0.0704 0.1346	116	0.0592 0.0578	126	0.0185 0.2011
97	0.1092 0.0194	107	0.2114 0.0452	117	0.1131 0.0531	127	0.1205 0.1855

Table 2.4.3.2.6.3-3. LSP Vector Quantization Table for LSPVQ3 (Part 1 of 2)

Index	(x,y)	Index	(x,y)	Index	(x,y)	Index	(x,y)
0	0.0255 0.0283	22	0.0952 0.0532	44	0.0513 0.1727	66	0.0621 0.0276
1	0.1296 0.0355	23	0.2393 0.0646	45	0.0711 0.2233	67	0.2183 0.0280
2	0.0543 0.0343	24	0.0490 0.0552	46	0.1085 0.0864	68	0.0311 0.1114
3	0.2073 0.0274	25	0.1619 0.0657	47	0.3398 0.0527	69	0.1382 0.0807
4	0.0204 0.1099	26	0.0845 0.0670	48	0.0414 0.0440	70	0.1284 0.0175
5	0.1562 0.0523	27	0.1784 0.2280	49	0.1356 0.0612	71	0.2605 0.0636
6	0.1388 0.0161	28	0.0191 0.1775	50	0.0964 0.0147	72	0.0230 0.0816
7	0.2784 0.0274	29	0.0272 0.2868	51	0.2173 0.0738	73	0.1739 0.0408
8	0.0112 0.0849	30	0.0942 0.0952	52	0.0465 0.1292	74	0.1074 0.0176
9	0.1870 0.0175	31	0.2628 0.1479	53	0.0877 0.1749	75	0.1619 0.1120
10	0.1189 0.0160	32	0.0278 0.0579	54	0.1104 0.0689	76	0.0784 0.1371
11	0.1490 0.1088	33	0.1565 0.0218	55	0.2105 0.1311	77	0.0448 0.3050
12	0.0969 0.1115	34	0.0814 0.0180	56	0.0580 0.0864	78	0.1189 0.0880
13	0.0659 0.3322	35	0.2379 0.0187	57	0.1895 0.0752	79	0.3039 0.1165
14	0.1158 0.1073	36	0.0276 0.1444	58	0.0652 0.0609	80	0.0424 0.0241
15	0.3183 0.1363	37	0.1199 0.1223	59	0.1485 0.1699	81	0.1672 0.0186
16	0.0517 0.0223	38	0.1200 0.0349	60	0.0514 0.1400	82	0.0815 0.0333
17	0.1740 0.0223	39	0.3009 0.0307	61	0.0386 0.2131	83	0.2432 0.0324
18	0.0704 0.0387	40	0.0312 0.0844	62	0.0933 0.0798	84	0.0584 0.1029
19	0.2637 0.0234	41	0.1898 0.0306	63	0.2473 0.0986	85	0.1137 0.1546
20	0.0692 0.1005	42	0.0863 0.0470	64	0.0334 0.0360	86	0.1015 0.0585
21	0.1287 0.1610	43	0.1685 0.1241	65	0.1375 0.0398	87	0.2198 0.0995

Table 2.4.3.2.6.3-3. LSP Vector Quantization Table for LSPVQ3 (Part 2 of 2)

Index	(x,y)	Index	(x,y)	Index	(x,y)	Index	(x,y)
88	0.0574 0.0581	98	0.0703 0.0216	108	0.0665 0.1799	118	0.1121 0.0555
89	0.1746 0.0647	99	0.2178 0.0482	109	0.0993 0.2213	119	0.1802 0.1509
90	0.0733 0.0740	100	0.0154 0.1421	110	0.1234 0.0631	120	0.0474 0.0886
91	0.1938 0.1737	101	0.1414 0.0994	111	0.3003 0.0762	121	0.1888 0.0610
92	0.0347 0.1710	102	0.1103 0.0352	112	0.0373 0.0620	122	0.0739 0.0585
93	0.0373 0.2429	103	0.3072 0.0473	113	0.1518 0.0425	123	0.1231 0.2379
94	0.0787 0.1061	104	0.0408 0.0819	114	0.0913 0.0300	124	0.0661 0.1335
95	0.2439 0.1438	105	0.2055 0.0168	115	0.1966 0.0836	125	0.0205 0.2211
96	0.0185 0.0536	106	0.0998 0.0354	116	0.0402 0.1185	126	0.0823 0.0822
97	0.1489 0.0178	107	0.1917 0.1140	117	0.0948 0.1385	127	0.2480 0.1179

Table 2.4.3.2.6.3-4. LSP Vector Quantization Table for LSPVQ4

Index	(x,y)	Index	(x,y)	Index	(x,y)	Index	(x,y)
0	0.0348 0.0311	16	0.0624 0.0228	32	0.0193 0.0596	48	0.0467 0.0348
1	0.0812 0.1145	17	0.1292 0.0979	33	0.1035 0.0957	49	0.1108 0.1048
2	0.0552 0.0461	18	0.0800 0.0195	34	0.0694 0.0397	50	0.0859 0.0306
3	0.1826 0.0263	19	0.2226 0.0285	35	0.1997 0.0253	51	0.1964 0.0463
4	0.0601 0.0675	20	0.0730 0.0862	36	0.0743 0.0603	52	0.0560 0.1013
5	0.1730 0.0172	21	0.1537 0.0601	37	0.1584 0.0321	53	0.1425 0.0533
6	0.1523 0.0193	22	0.1115 0.0509	38	0.1346 0.0346	54	0.1142 0.0634
7	0.2449 0.0277	23	0.2720 0.0354	39	0.2221 0.0708	55	0.2391 0.0879
8	0.0334 0.0668	24	0.0218 0.1167	40	0.0451 0.0732	56	0.0397 0.1084
9	0.0805 0.1441	25	0.1212 0.1538	41	0.1040 0.1415	57	0.1345 0.1700
10	0.1319 0.0207	26	0.1074 0.0247	42	0.1184 0.0230	58	0.0976 0.0248
11	0.1684 0.0910	27	0.1674 0.1710	43	0.1853 0.0919	59	0.1887 0.1189
12	0.0582 0.1318	28	0.0322 0.2142	44	0.0310 0.1661	60	0.0644 0.2087
13	0.1403 0.1098	29	0.1263 0.0777	45	0.1625 0.0706	61	0.1262 0.0603
14	0.0979 0.0832	30	0.0981 0.0556	46	0.0856 0.0843	62	0.0877 0.0550
15	0.2700 0.1359	31	0.2119 0.1710	47	0.2902 0.0702	63	0.2203 0.1307

Table 2.4.3.2.6.3-5. LSP Vector Quantization Table for LSPVQ5

Index	(x,y)	Index	(x,y)	Index	(x,y)	Index	(x,y)
0	0.0360 0.0222	16	0.0570 0.0180	32	0.0210 0.0478	48	0.0443 0.0334
1	0.0820 0.1097	17	0.1135 0.1382	33	0.1029 0.1020	49	0.0835 0.1465
2	0.0601 0.0319	18	0.0778 0.0256	34	0.0722 0.0181	50	0.0912 0.0138
3	0.1656 0.0198	19	0.1901 0.0179	35	0.1730 0.0251	51	0.1716 0.0442
4	0.0604 0.0513	20	0.0807 0.0622	36	0.0730 0.0488	52	0.0620 0.0778
5	0.1552 0.0141	21	0.1461 0.0458	37	0.1465 0.0293	53	0.1316 0.0450
6	0.1391 0.0155	22	0.1231 0.0178	38	0.1303 0.0326	54	0.1186 0.0335
7	0.2474 0.0261	23	0.2028 0.0821	39	0.2595 0.0387	55	0.1446 0.1665
8	0.0269 0.0785	24	0.0387 0.0927	40	0.0458 0.0584	56	0.0486 0.1050
9	0.1463 0.0646	25	0.1496 0.1004	41	0.1569 0.0742	57	0.1675 0.1019
10	0.1123 0.0191	26	0.0888 0.0392	42	0.1029 0.0173	58	0.0880 0.0278
11	0.2015 0.0223	27	0.2246 0.0341	43	0.1910 0.0495	59	0.2214 0.0202
12	0.0785 0.0844	28	0.0295 0.1462	44	0.0605 0.1159	60	0.0539 0.1564
13	0.1202 0.1011	29	0.1156 0.0694	45	0.1268 0.0719	61	0.1142 0.0533
14	0.0980 0.0807	30	0.1022 0.0473	46	0.0973 0.0646	62	0.0984 0.0391
15	0.3014 0.0793	31	0.2226 0.1364	47	0.2872 0.0428	63	0.2130 0.1089

2.4.3.2.7 Converting the LSP Frequencies to Transmission Codes for Rate 1/8

For Rate 1/8 frames, the LSP conversion process is shown in Figure 2.4.3.2.7-1.

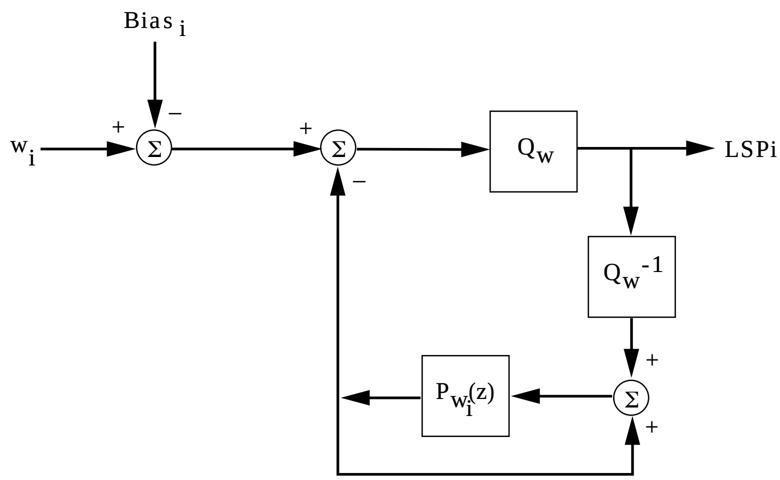


Figure 2.4.3.2.7-1. Converting the LSP Frequencies to Transmission Codes for Rate 1/8

Each of the ten LSP frequencies centers roughly around a bias value (the frequencies equal the bias values when the input speech has flat spectral characteristics and no formant prediction can be performed). The bias used for each LSP frequency is

$$Bias_i = \frac{i}{P+1}, \qquad 1 \le i \le 10 \tag{2.4.3.2.7-1}$$

where P is equal to 10.

The predictor $P_{w_i}(z)$ is

$$P_{w_i}(z) = 0.90625 z^{-1} \tag{2.4.3.2.7-2}$$

The state of the LSP predictor is updated once per frame, unless a Blank packet has been requested. There is one predictor for each LSP frequency.

The one-bit quantizer used for Rate 1/8 encoding Q_W, for the ith LSP frequency is a linear quantizer which is the same for all ten LSP frequencies. Each LSP frequency is quantized as

$$Q_w(x) = \begin{cases} 0, & \text{if } x < 0 \\ 1, & \text{if } x \ge 0 \end{cases} \tag{2.4.3.2.7-3}$$

2.4.3.3 Decoding LSP Frequencies and Converting to LPC Coefficients

The decoding process consists of the following steps:

- Convert the LSP transmission codes to LSP frequencies
- Check the stability of the LSP frequencies for Rate 1/8 Encoding
- Low-pass filter the LSP frequencies
- Interpolate the LSP frequencies
- Convert the interpolated LSP frequencies to LPC coefficients
- Scale the LPC coefficients to perform bandwidth expansion.
- Update state for predictor memory $Pw_i(z)$

The steps taken by the receiving decoder (see 2.4.11.2) are similar to those taken by the transmitting speech codec, unless a packet type equal to insufficient frame quality is received (see 2.3.2.2).

2.4.3.3.1 Converting the LSP Transmission Codes to LSP Frequencies

The LSPs are decoded at both the transmitting encoder and the receiving decoder. First, the LSP codes are used to regenerate the quantized LSP frequencies, wq_i.

For Rate 1, Rate 1/2, and Rate 1/4, the quantized LSP frequencies can be reconstructed with the following pseudocode (see 2.4.3.2.6.2).

```
{
        wq1 = Lkbst(1)(1,1)
        wq2 =  wq1+Lkbst(1)(1,2)
        for i =  2 to  5{
            wq(2i-1) = wq(2i-2) + Lkbst(i)(i,1)
            wq(2i) = wq(2i-1) + Lkbst(i)(i,2)
        }
}
```

Figure 2.4.3.3.1-1 describes the LSP frequency regeneration process for Rate 1/8 and insufficient frame quality frames.

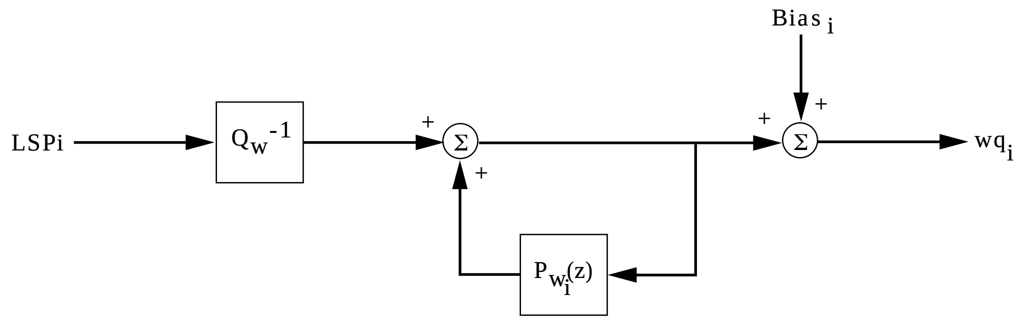


Figure 2.4.3.3.1-1. Converting the LSP Transmission Codes to LSP Frequencies for Rate 1/8 and Insufficient Frame Quality Frames

The predictor $P_{w_i}(z)$ is the same as in Equation 2.4.3.2.7-2. The state of the predictor is updated for every packet except for a Blank packet. The bias is given in Equation 2.4.3.2.7-1. The one-bit inverse quantizer Q_w^{-1} is

$$Q_w^{-1} = \begin{cases} -0.02, & \text{if LSPi} = 0 \\ 0.02, & \text{if LSPi} = 1 \end{cases} \qquad (2.4.3.3.1\text{-}1)$$

2.4.3.3.2 Checking the Stability of the LSP Frequencies for Rate 1/8 Encoding

Before converting the LSP frequencies back to LPC coefficients for Rate 1/8 frames, the LSP frequencies are checked to ensure that the resulting LPC filter is stable. Quantization noise or channel errors in LSP frequencies may result in an unstable LPC filter. Stability is guaranteed if the LSP frequencies remain ordered. In addition, the LSP frequencies are forced to be at least 80 Hz apart, so as to prevent unusually large peaks in the formant synthesis filter response. This ordering and minimum spacing are enforced using the following algorithm

```
{
  wq0 = 0.0
  i = 0
  while (i < 10)
     {
       if  ( (wqi+1 -wqi) < wqmin)
          wqi+1 = wqi + wqmin
       i = i + 1
     }
  wq11 = 1.0
  while (i > 0)
     {
       if  ( (wqi+1 - wqi) < wqmin)
          wqi = wqi+1 - wqmin
       i = i - 1
     }
}
```

A wq_{min} of 0.02 is used, which results in 80 Hz separation between LSP frequencies.

2.4.3.3.3 Low-Pass Filtering the LSP Frequencies

The low-pass filtered LSP frequencies $\hat{w}_i$ are given by

$$\hat{w}_i(\text{current frame}) = SM\,\hat{w}_i(\text{previous frame}) + (1 - SM)wq_i(\text{current frame}) \qquad (2.4.3.3.3\text{-}1)$$

where the value of SM depends on the packet rate.

This reduces quantization noise effects in Rate 1/8 and insufficient frame quality (erasure) packets. For both the encoder and decoder, a counter is used to track the number of consecutive Rate 1/8 packets. If the current packet is Rate 1/8, the counter is incremented. If the current packet is either Rate 1, Rate 1/2, or Rate 1/4, the counter is set to zero. For insufficient frame quality (erasure) packets the counter is unchanged. The value of SM that is used in Equation 2.4.3.3.3-1 is given by

$$SM = \begin{cases} 0, & \text{if packet is Rate 1, 1/2, or 1/4} \\ 0.125, & \text{if packet is Rate 1/8 and counter} < 10 \\ 0.9, & \text{if packet is Rate 1/8 and counter} \geq 10 \\ 0.875, & \text{if an insufficient frame quality (erasure) packet} \end{cases} \qquad (2.4.3.3.3\text{-}2)$$

2.4.3.3.4 Interpolating the LSP Frequencies

The LSP frequencies are interpolated for each subframe of the pitch or codebook search, depending on the selected rate.

In calculating the original LPC coefficients, a speech window centered between the 139th and 140th samples of the frame was used. In performing the pitch and codebook searches for Rate 1 and Rate 1/2 packets subframes, LPC coefficients which are accurate at the center of the particular pitch subframe should be used. For Rate 1/8, LPC coefficients which are accurate at the center of the single codebook subframe should be used. For Rate 1/4, LPC coefficients which are accurate at the center of four 40-sample subframes should be used. These LPC coefficients are approximated by interpolating between the previous frame's and the current frame's LSP frequencies, and then converting the resulting interpolated LSP frequencies back into LPC coefficients.

The exact interpolation used for each subframe of each rate is shown in Table 2.4.3.3.4-1. In all cases $\hat{w}_i$ (previous) is the ith filtered LSP frequency from the previous frame and $\hat{w}_i$ (current) is the ith filtered LSP frequency from the current frame.

Table 2.4.3.3.4-1. LSP Subframe Interpolation for All Rates

Rate 1, Rate 1/2, and Rate 1/4	For Pitch or 40-Sample Subframe
$\hat{w}'_i = 0.75\,\hat{w}_i\,(\text{previous}) + 0.25\,\hat{w}_i\,(\text{current})$	1
$\hat{w}'_i = 0.5\,\hat{w}_i\,(\text{previous}) + 0.5\,\hat{w}_i\,(\text{current})$	2
$\hat{w}'_i = 0.25\,\hat{w}_i\,(\text{previous}) + 0.75\,\hat{w}_i\,(\text{current})$	3
$\hat{w}'_i = \hat{w}_i\,(\text{current})$	4

Rate 1/8	For Codebook Subframe
$\hat{w}'_i = 0.375\,\hat{w}_i\,(\text{previous}) + 0.625\,\hat{w}_i\,(\text{current})$	1

Insufficient Frame Quality Packet (Erasure)	For Codebook Subframe
$\hat{w}'_i = \hat{w}_i\,(\text{previous})$	1

2.4.3.3.5 Converting the Interpolated LSP Frequencies to LPC Coefficients

The interpolated LSP frequencies are converted to LPC coefficients which are used by the receiving decoder for speech generation as described in 2.4.11.2. In addition, the LPC coefficients are used in the pitch and codebook searches. The conversion method is described in the following.

Compute $\hat{P}_A(z)$ and $\hat{Q}_A(z)$ from the LSP frequencies using

$$\hat{P}_A(z) = \left(1 + z^{-1}\right)\prod_{j=1}^{5}\left(1 - 2z^{-1}\cos\left(\pi\,\hat{w}'_{(2j-1)}\right) + z^{-2}\right) \tag{2.4.3.3.5-1}$$

and

$$\hat{Q}_A(z) = \left(1 - z^{-1}\right)\prod_{j=1}^{5}\left(1 - 2z^{-1}\cos\left(\pi\,\hat{w}'_{(2j)}\right) + z^{-2}\right) \tag{2.4.3.3.5-2}$$

The LPC coefficients are computed from the coefficients of $\hat{P}_A(z)$ and $\hat{Q}_A(z)$ using

$$
\begin{aligned}
A(z) &= \frac{\hat{P}_A(z) + \hat{Q}_A(z)}{2} \\
&= 1 + \frac{\left(\hat{p}_1 + \hat{q}_1\right)}{2} z^{-1} + \ldots + \frac{\left(\hat{p}_5 + \hat{q}_5\right)}{2} z^{-5} + \frac{\left(\hat{p}_5 - \hat{q}_5\right)}{2} z^{-6} + \ldots + \frac{\left(\hat{p}_1 - \hat{q}_1\right)}{2} z^{-10} \\
&= 1 - a'_1 z^{-1} \ldots - a'_{10} z^{-10}
\end{aligned}
\qquad (2.4.3.3.5\text{-}3)
$$

so

$$
a'_i = \begin{cases} -\dfrac{\hat{p}_i + \hat{q}_i}{2}, & 1 \le i \le 5 \\ -\dfrac{\hat{p}_{11-i} - \hat{q}_{11-i}}{2}, & 6 \le i \le 10 \end{cases}
\qquad (2.4.3.3.5\text{-}4)
$$

The LPC coefficients for the particular subframe are the a'_i given in Equation 2.4.3.3.5-4.

2.4.3.3.6 Scaling the LPC Coefficients to Perform Bandwidth Expansion

After converting the interpolated LSP coefficients to LPC coefficients, the LPC coefficients are scaled to perform bandwidth expansion. Each LPC coefficient, a'_i, is scaled by β^i (β to the ith power) as

$$
\hat{a}_i = \beta^i a'_i, \qquad 1 \le i \le P \qquad (2.4.3.3.6\text{-}1)
$$

where β is 0.9883.

2.4.4 Determining the Packet Type (Rate)

The determination of the packet type is performed in two stages as shown in Figure 2.4.4-1. In the first stage of the rate determination algorithm (RDA) a voice activity decision is made using a multiband energy thresholding scheme in each band. This voice activity detection decides if the current frame should be encoded at Rate 1, Rate 1/2, or Rate 1/8.

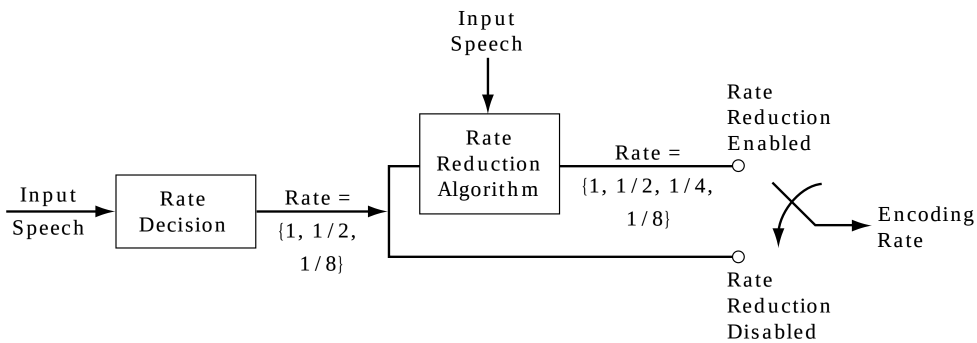


Figure 2.4.4-1. Two Stages in the Rate Determination Algorithm

When rate reduction is enabled as defined in 2.3.5, the second stage of the RDA decides if the current frame should be encoded at a reduced rate. The valid rate modifications for the second stage are listed in Table 2.4.4-1. Rate 1/8 frames are left unchanged. Rate 1 is used for transitional, reduced periodicity or poorly modeled frames in which the highest encoding rate is necessary to achieve good speech quality. Rate 1/2 is used for well-modeled, stationary and periodic frames. Rate 1/4 is used for unvoiced speech.

Table 2.4.4-1. Valid Rate Modifications for the Rate Reduction Algorithm

Input Rate	Output Rate
1	1, 1/2, 1/4
1/2	1/2, 1/4
1/8	1/8

Six features are used by the second stage of the rate determination algorithm to detect the encoding modes. Zero crossings and the normalized autocorrelation function (NACF) are used to make the voiced/unvoiced classifications. A target SNR is used to estimate the coding efficiency of the CELP model. The differential prediction gain, differential LSP, and NACF are used to detect stationary and transitional speech characteristics. Finally, an average-frame-energy to current-frame-energy ratio and a differential LSP are used to detect temporally masked speech frames.

Thresholds on these six features are used to detect the various encoding modes. Thresholds are adapted based on background noise energy and the desired average encoding rate for active speech.

2.4.4.1 First Stage of Rate Determination Algorithm

The first stage of the rate determination algorithm is used to classify the input speech as active speech or background noise. One of three encoding rates are selected in this stage: Rate 1, Rate 1/2, and Rate 1/8. Active speech is encoded at Rate 1 or Rate 1/2, and background noise is encoded at Rate 1/8.

2.4.4.1.1 Computing Band Energy

The rate determination algorithm uses energy thresholds to determine the encoding rate for the current frame. The input speech is divided into two bands: band f(1) spans 0.3-2.0 kHz, band f(2) spans 2.0-4.0 kHz.[6] The band energy for band f(i), $BE_{f(i)}$, is calculated as

$$BE_{f(i)} = R(0)R_{f(i)}(0) + 2.0\sum_{m=1}^{L_h-1} R(m)R_{f(i)}(m) \qquad (2.4.4.1.1\text{-}1)$$

where

$$R_{f(i)}(k) = \sum_{m=0}^{L_h-1-k} h_i(m)h_i(m+k) \qquad (2.4.4.1.1\text{-}2)$$

where $h_i(k)$ is the impulse response of the band-pass filter i, R(k) is the autocorrelation sequence defined in Equation 2.4.3.2.3-1, and L_h is the length of the impulse response of the band-pass filters.

The band-pass filters used for both frequency bands are defined in Table 2.4.4.1.1-1.

[6]Whenever a variable (or symbol or value) with a subscript f(i) appears in any equation or pseudocode in 2.4.4, it refers to a variable (or symbol or value) associated with either band f(1) or band f(2).

Table 2.4.4.1.1-1. FIR Filter Coefficients Used for Band Energy Calculations

k	$h_1(k)$ (lower band)	k	$h_2(k)$ (upper band)
0	-5.557699E-02	0	-1.229538E-02
1	-7.216371E-02	1	4.376551E-02
2	-1.036934E-02	2	1.238467E-02
3	2.344730E-02	3	-6.243877E-02
4	-6.071820E-02	4	-1.244865E-02
5	-1.398958E-01	5	1.053678E-01
6	-1.225667E-02	6	1.248720E-02
7	2.799153E-01	7	-3.180645E-01
8	4.375000E-01	8	4.875000E-01
9	2.799153E-01	9	-3.180645E-01
10	-1.225667E-02	10	1.248720E-02
11	-1.398958E-01	11	1.053678E-01
12	-6.071820E-02	12	-1.244865E-02
13	2.344730E-02	13	-6.243877E-02
14	-1.036934E-02	14	1.238467E-02
15	-7.216371E-02	15	4.376551E-02
16	-5.557699E-02	16	-1.229538E-02

2.4.4.1.2 Calculating Rate Determination Thresholds

The rate determination thresholds for each frequency band f(i) are a function of both the background noise estimate, $B_{f(i)}(k-1)$, and the estimated signal-to-noise ratio, $S_{f(i)}(k-1)$, of the previous or (k-1)th frame. Two thresholds for each band are computed as

$$T_1\left(B_{f(i)}(k-1), SNR_{f(i)}(k-1)\right) = k1\left(SNR_{f(i)}(k-1)\right) B_{f(i)}(k-1) \quad (2.4.4.1.2\text{-}1)$$

$$T_2\left(B_{f(i)}(k-1), SNR_{f(i)}(k-1)\right) = k2\left(SNR_{f(i)}(k-1)\right) B_{f(i)}(k-1) \quad (2.4.4.1.2\text{-}2)$$

where the integer $SNR_{f(i)}(k-1)$ is

$$SNR_{f(i)}(k-1) = \begin{cases} 0, & QSNRU_{f(i)}(k-1) < 0 \\ QSNRU_{f(i)}(k-1), & 0 \le QSNRU_{f(i)}(k-1) \le 7 \\ 7, & QSNRU_{f(i)}(k-1) > 7 \end{cases} \quad (2.4.4.1.2\text{-}3)$$

where

$$QSNRU_{f(i)}(k-1) = \text{round}\left((10\log_{10}\left(S_{f(i)}(k-1) / B_{f(i)}(k-1)\right) - 20) / 5\right) \quad (2.4.4.1.2\text{-}4)$$

k1(•) and k2(•) are functions defined in Table 2.4.4.1.2-1, and $B_{f(i)}(k-1)$ and $S_{f(i)}(k-1)$ are defined in 2.4.4.2.2 and 2.4.4.2.3, respectively.

Table 2.4.4.1.2-1. Threshold Scale Factors as a Function of SNR

$SNR_{f(i)}(k-1)$	$k1(SNR_{f(i)}(k-1))$	$k2(SNR_{f(i)}(k-1))$
0	7.0	9.0
1	7.0	12.6
2	8.0	17.0
3	8.6	18.5
4	8.9	19.4
5	9.4	20.9
6	11.0	25.5
7	15.8	39.8

The threshold scale factors are identical for the low- and high-frequency bands.

2.4.4.1.3 Comparing Thresholds

For each of the two frequency bands, the two thresholds required to select among Rate 1, Rate 1/2, or Rate 1/8 are maintained as described in 2.4.4.1.2.

Band energy, $BE_{f(i)}$, is compared with two thresholds: $T_1(B_{f(i)}(k-1), SNR_{f(i)}(k-1))$ and $T_2(B_{f(i)}(k-1), SNR_{f(i)}(k-1))$. If $BE_{f(i)}$ is greater than both thresholds, Rate 1 is selected. If $BE_{f(i)}$ is greater than only one threshold, Rate 1/2 is selected. If $BE_{f(i)}$ is below both thresholds, Rate 1/8 is selected. This procedure is performed for both frequency bands and the higher of the two encoding rates selected from the individual bands is chosen as the encoding rate of the current frame k.

2.4.4.1.4 Performing Hangover

If the last frame's encoding rate was Rate 1 and the current frame is determined not to be a Rate 1 frame, then the next M frames are encoded as Rate 1 before allowing the encoding rate to drop to Rate 1/2 and finally to Rate 1/8. The number of hangover frames, M, is a function of the $SNR_{f(1)}(k-1)$ (the SNR in the lower frequency band) and is denoted as Hangover($SNR_{f(1)}(k-1)$ in Table 2.4.4.1.4-1. $SNR_{f(1)}(k-1)$ is calculated as defined in Equation 2.4.4.1.2-3. The hangover algorithm is defined by the following pseudocode:

```
{
  if (Rate1(k) == Rate 1) count = 0
  if (Rate1(k-1) == Rate 1 and Rate1(k) != Rate 1){
     if(count < M){
       Rate1(k) = Rate 1
       count = count + 1
     }
  }
}
```

where $Rate_1(k)$ and $Rate_1(k-1)$ are the rates of the current and previous frame after the first stage of the RDA, respectively.

Table 2.4.4.1.4-1. Hangover Frames as a Function of SNR

$SNR_{f(1)}(k-1)$	$Hangover(SNR_{f(1)}(k-1))$
0	7
1	7
2	7
3	3
4	0
5	0
6	0
7	0

2.4.4.1.5 Constraining Rate Selection

The rate selected by the procedures described in 2.4.4.1.3 and 2.4.4.1.4 is used for the current frame except where it is modified by the following constraints:

If, by the first stage of the RDA, the previous frame was selected as Rate 1 and the current frame is selected as Rate 1/8, then the encoding rate of the current frame should be modified to Rate 1/2. There are no other restrictions on encoding rate transitions for the first stage of the RDA.

If the speech codec has been commanded not to generate a Rate 1 packet and the rate determined by the first and second stage of the RDA is Rate 1, it generates a Rate 1/2 packet. If the speech codec has been told to generate a Blank packet, it generates a Blank packet regardless of the rate determined by the two stages of the RDA. If the codec is operating in reduced rate mode, the encoding rate selected by the first stage of RDA is modified as described in 2.4.4.3.

2.4.4.2 Updating Smoothed Band Energy

After the first stage of RDA is complete, RDA parameters should be updated as described in 2.4.4.2.1 through 2.4.4.2.3.

2.4.4.2.1 Updating the Smoothed Band Energy

The band energy, $BE_{f(i)}$, calculated in Equation 2.4.4.1.1-1 is smoothed and used to estimate both the background noise energy (see 2.4.4.2.2) and signal energy (see 2.4.4.2.3) in each band. The smoothed band energy, $E^{sm}_{f(i)}(k)$, is computed as

$$E^{sm}_{f(i)}(k) = 0.6\ E^{sm}_{f(i)}\left(k-1\right) + 0.4\ BE_{f(i)} \qquad (2.4.4.2.1\text{-}1)$$

with initial conditions

$$E^{sm}_{f(1)}\left(0\right) = 3200000$$
$$E^{sm}_{f(2)}\left(0\right) = 320000$$

where k refers to the current frame.

2.4.4.2.2 Updating Background Noise Estimate

To update the background noise and signal energy estimates, the normalized autocorrelation function is computed on the decimated prediction residual, $e_d(n)$. $e_d(n)$ is obtained by low-pass filtering and decimating by a factor of two the prediction residual, e(n), as shown in Figure 2.4.4.2.2-1. This reduces the complexity of the NACF calculation. An example of a low-pass filter used in the decimation process is given in Table 2.4.4.2.2-1.

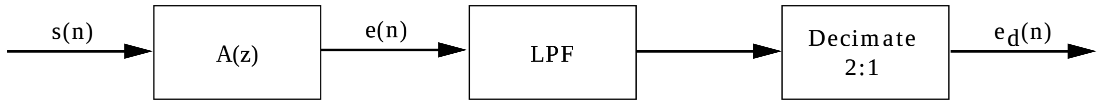


Figure 2.4.4.2.2-1. Decimation of the Prediction Residual for NACF Computation

The formant prediction filter, A(z), used in Figure 2.4.4.2.2-1 should be the same formant prediction filter used in the analysis-by-synthesis encoding procedure described in 2.4.5.1. More specifically, A(z) will be the same filter as defined in the numerator of Equation 2.4.5.1-2. Also, the reconstructed LPC coefficients that define A(z) will be known at this stage of the RDA. After the second stage of RDA, the encoding rate is either Rate 1/8 or one of the three active speech encoding rates of Rate 1, Rate 1/2, or Rate 1/4. The LSP quantizer, the low-pass filtering, and the interpolation process for Rate 1, Rate 1/2, and Rate 1/4 frames are identical.

Table 2.4.4.2.2-1. Impulse Response of LPF Used in the Decimation Process to Calculate the NACF

n	$h_d(n)$	n	$h_d(n)$
0	2.725341E-03	9	2.767512E-01
1	1.028254E-02	10	1.166278E-01
2	5.973260E-03	11	-1.323563E-02
3	-2.308975E-02	12	-5.009796E-02
4	-5.009796E-02	13	-2.308975E-02
5	-1.323563E-02	14	5.973260E-03
6	1.166278E-01	15	1.028254E-02
7	2.767512E-01	16	2.725341E-03
8	3.500000E-01		

The normalized autocorrelation function, NACF, is computed as

$$\mathrm{NACF} = \frac{\max\limits_{T \in \{10,11,\ldots,59\}} \left\{ \sum_{m=0}^{N_d-1} e_d(m)\, e_d(m-T) \right\}}{\left\{ \text{Energy in } e_d(T) \right\}} \tag{2.4.4.2.2-1}$$

where $N_d = 80$ and

$$\text{Energy in } e_d(T_{max}) = 0.5 \sum_{n=0}^{N_d-1} \left\{ e_d^2(n) + e_d^2(n - T_{max}) \right\}$$

The Energy in $e_d(T)$ is computed after the T that maximizes the numerator in Equation 2.4.4.2.2-1, T_{max}, is computed.

An estimate of the background noise level, $B_{f(i)}(k)$, is computed for the current, or kth, frame using $B_{f(i)}(k-1)$, $E^{sm}{}_{f(i)}(k)$ (see 2.4.4.2.1) and $SNR_{f(i)}(k-1)$ (see 2.4.4.1.2). Pseudocode describing the background noise update for band f(i) is given as

```
{
   if(NACF <  0.38 for 8 or more consecutive frames)
      Bf(i)(k)= min (Esmf(i)(k), 5059644, max (1.03Bf(i)(k-1), Bf(i)(k-1)+ 1))
   else  {
      if( SNRf(i)(k-1) > 3)
        Bf(i)(k)= min (Esmf(i)(k), 5059644, max (1.00547Bf(i)(k-1), Bf(i)(k-1)+ 1))
      else
        Bf(i)(k)= min (Esmf(i)(k), 5059644, Bf(i)(k-1))
   }
   if(Bf(i)(k) < lownoise(i)) Bf(i)(k) = lownoise(i)
}
```

where NACF, $E^{sm}{}_{f(i)}(k)$, and $SNR_{f(i)}(k-1)$ are defined in Equations 2.4.4.2.2-1, 2.4.4.2.1-1, and 2.4.4.1.2-3, respectively, lownoise(1) equals 10.0, and lownoise(2) equals 5.0.

At initialization, the background noise estimate for the first frame, $B_{f(i)}(0)$, is set to 5059644 for both frequency bands. If the audio input to the encoder is disabled and then enabled, the background noise estimate and the NACF threshold counters are initialized.[7]

2.4.4.2.3 Updating Signal Energy Estimate

The signal energy, $S_{f(i)}(k)$, is computed as

```
{
If(NACF > 0.5 for 5 or more consecutive frames)
  Sf(i)(k)= max(Esmf(i)(k), 0.97 Sf(i)(k-1))
else
  Sf(i)(k)= max(Esmf(i)(k), Sf(i)(k-1))
}
```

where NACF and $E^{sm}_{f(i)}(k)$ are defined in Equations 2.4.4.2.2-1 and 2.4.4.2.1-1, respectively.

At initialization, the signal energy estimates for the first frame, $S_{f(1)}(0)$ and $S_{f(2)}(0)$, are set to 3200000 and 320000, respectively. If the audio input to the encoder is disabled and then enabled, the signal energy estimate and the NACF threshold counters are initialized.

2.4.4.3 Second Stage of Rate Determination Algorithm: Rate Reduction

If the codec is operating with rate reduction enabled and the first stage of the RDA detects a Rate 1 or Rate 1/2 frame, then the second stage in the RDA is used to identify the most efficient encoding rate based upon statistics of the input speech. The second stage maximizes voice quality while constraining the average rate to a desired target. This is necessary for increasing the system capacity for heavily loaded conditions. The service option control order required to enable rate reduction is defined in Table 2.3.5.2-2. This table defines five average rate target values that control the RDA when rate reduction is enabled.

To efficiently achieve the average encoding rate, the selected encoding rate is matched to the mode or characteristic of the input speech. Figure 2.4.4.3-1 is a flowchart showing the modes being selected and the encoding rates being used. RRM Level denotes Reduced Rate Mode Level (see Table 2.3.5.2-2).

---

[7]This prevents the silence before the audio is connected from being mistaken as unusually low background noise.

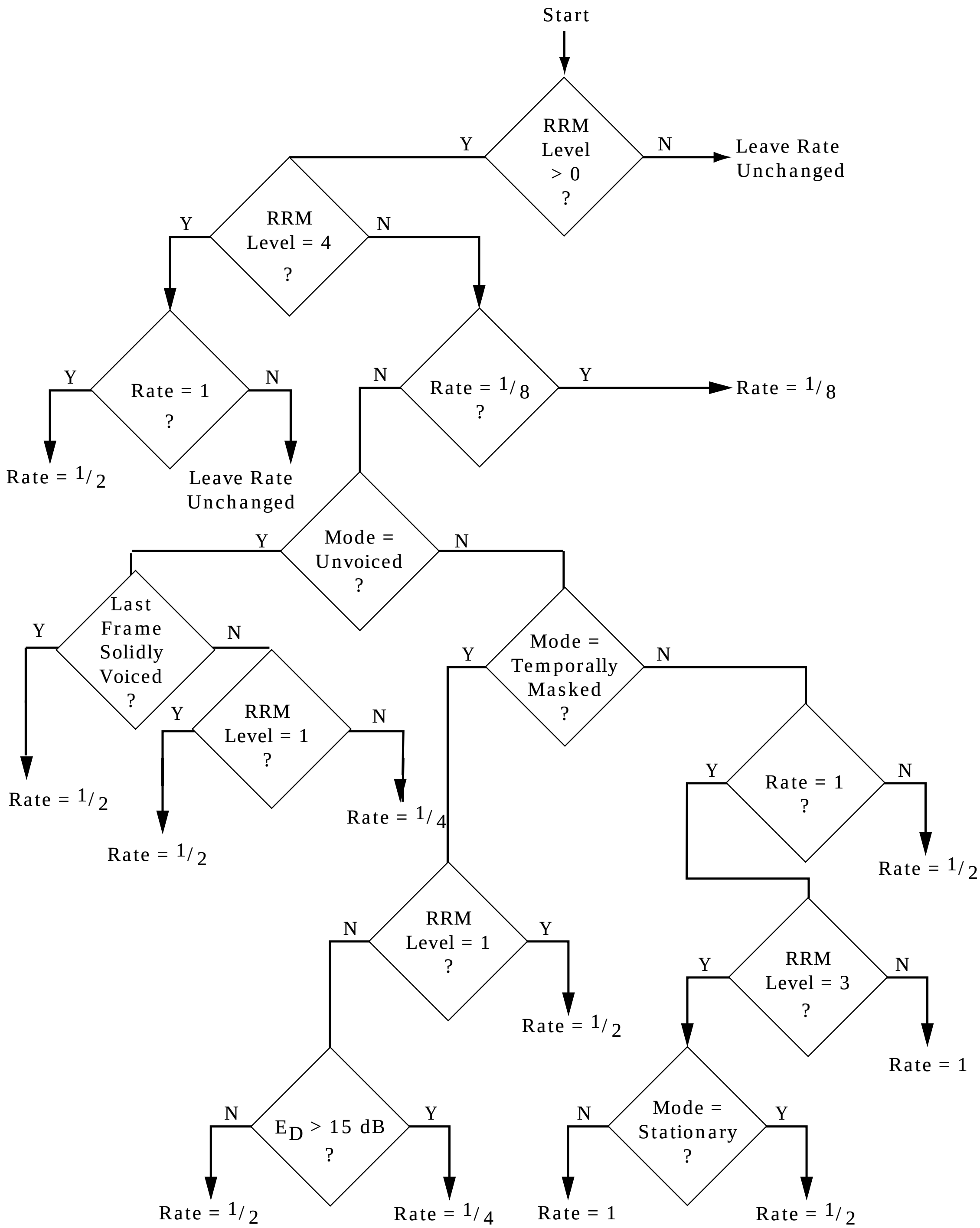


Figure 2.4.4.3-1. Flowchart for the Second Stage of the Rate Determination Algorithm

The input speech signal is classified into four modes: non-stationary voiced, stationary voiced, unvoiced, and temporally masked speech. Six features are used to make the mode selection. Zero crossings and the normalized autocorrelation function (NACF) are used to make the voiced/unvoiced classification. Zero crossings of the speech signal are defined as the number of sign changes in the input speech signal after the DC component has been removed. A procedure for calculating the number of zero crossings follows:

```
{
  zero_cross = 0;
  for(n=0; n < LA-1; n++)
     if(s(n)*s(n+1) < 0) zero_cross++;
}
```

where s(n) is defined in 2.4.3.2.2. The NACF function is defined in Equation 2.4.4.2.2-1.

A target SNR feature, Target_SNR, is used to estimate the coding accuracy of the CELP model and thus to distinguish stationary versus non-stationary speech. It is defined as

$$\text{Target\_SNR} = 10\log_{10}\left(E_T / E_{TMN}\right) \tag{2.4.4.3-1}$$

where

$$E_T = \sum_{n=0}^{L_A-1} x^2(n), \tag{2.4.4.3-2}$$

where x(n) is the target signal defined in Table 2.4.5.1.1-1, where

$$E_{TMN} = \sum_{n=0}^{L_A-1} \left(err(n)\right)^2 \tag{2.4.4.3-3}$$

and where err(n) is the encoding error signal defined in Figure 2.4.8-1.

Note that Target_SNR is computed on the previously encoded speech frame and is used as a feature in the current frame for rate selection.

The differential prediction gain, differential LSP, and NACF are also used to distinguish stationary from non-stationary speech states. The differential features are calculated using values from the current frame k and the previous frame k-1. The differential prediction gain is defined as

$$\Delta Pg(k) = 10\ \log_{10}\left(\frac{Pg(k)}{Pg(k-1)}\right) \tag{2.4.4.3-4}$$

where

$$Pg(k) = \frac{R(0)}{\left(R(0) - \sum_{i=1}^{P} a_i R(i)\right)} \quad (2.4.4.3\text{-}5)$$

with R(i) and a_i are defined in 2.4.3.2.3 and 2.4.3.2.4, respectively.

The differential LSP is defined as

$$\Delta LSP(k) = \sum_{i=1}^{P} \left(w_i(k) - w_i(k-1)\right)^2 \quad (2.4.4.3\text{-}6)$$

with $w_i(k)$ defined in 2.4.3.2.5.

The average-frame-energy to current-frame-energy ratio and the differential LSP are used to detect temporally masked speech frames that can be coded at reduced encoding rates, either Rate 1/2 or Rate 1/4. The average-frame-energy to current-frame-energy ratio is defined as

$$E_D(k) = E_{AVG}(k) - 10\ \log_{10}(R_D) \quad (2.4.4.3\text{-}7)$$

where

$$E_{AVG}(k) = \lambda E_{AVG}(k-1) + (1-\lambda)10\ \log_{10}(R_D) \quad (2.4.4.3\text{-}8)$$

and $\lambda = 0.8825$.

And

$$R_D = .625\ R(0)\ +\ .375\ R(0)\_\text{previous} \quad (2.4.4.3\text{-}9)$$

where R(0)_previous is the R(0) value from the previous frame.

Since the autocorrelation function, the LSP parameters, and the prediction gain used in Equations 2.4.4.3-4, 2.4.4.3-6, and 2.4.4.3-7 are defined for the offset formant-synthesis-filter analysis window used in Equations 2.4.3.2.2-1 and 2.4.3.2.3-1, an interpolation is used to adjust these features to the non-offset analysis window used for encoding. The interpolated features PG_D and LSP_D are defined as

$$PG_D = 0.625\ \Delta Pg(k) + 0.375\ \Delta Pg(k-1) \quad (2.4.4.3\text{-}10)$$

$$LSP_D = 0.625\ \Delta LSP(k) + 0.375\ \Delta LSP(k-1) \quad (2.4.4.3\text{-}11)$$

If the audio input to the encoder is disabled and is then enabled, all the features that require “last frame” estimates for interpolation purposes should be initialized to zero.

2.4.4.3.1 Unvoiced Detection

An input frame is declared either voiced or unvoiced as a function of the NACF and zero crossings features. The voiced/unvoiced classification is made according to the following pseudocode:

```
{
  discriminant = c(0)*NACF + c(1)*zero_cross + c(2)
  if((NACF > 0.5 and zero_cross < 80)
  or (NACF < 0.25 and zero_cross < 45)
  or (Pg(k) > 15.0))
     mode = voiced
  else if (discriminant ≥ 0.0)
     mode = voiced
  else
     mode = unvoiced
}
```

where the coefficients c(i) in computing the discriminant are defined as c(0)=5.190283, c(1)=-0.092413, and c(2)=3.091836. NACF is defined in Equation 2.4.4.2.2-1 and zero_cross is defined in 2.4.4.3. A speech frame is solidly voiced if NACF ≥ 0.5 and zero_cross < 60. The unvoiced encoding rate may be modified if the previous frame was determined to be solidly voiced. This rate modification is described by the following pseudocode:

```
{
  if((mode == unvoiced) and (NACF_last ≥ 0.5) and (zero_cross_last < 60))
     rate = 1/2                    /* Previous frame was solidly voiced */
  else if ((mode == unvoiced) and ((RRM_Level == 2) or (RRM_Level == 3)))
     rate = 1/4
  else if ((mode == unvoiced) and ((RRM_Level == 1) or (RRM_Level == 4)))
     rate = 1/2
  NACF_last = NACF
  rate_last = rate
  zero_cross_last = zero_cross
  if(rate == 1/8) NACF_last = 0, zero_cross_last = 100
}
```

If the previous frame was not solidly voiced, then the encoding rate of the current unvoiced frame is chosen as a function of the reduced rate level and is given in Table 2.4.4.3.1-1.

Table 2.4.4.3.1-1. Unvoiced Encoding Rate as a Function of Reduced Rate Level

Reduced Rate Level as Defined in Table 2.3.5.2-1	Unvoiced Encoding Rate
0	Rate 1
1	Rate 1/2
2	Rate 1/4
3	Rate 1/4
4	Rate 1/2

2.4.4.3.2 Temporally Masked Frame Detection

Temporally masked frames are defined as frames of speech in which the signal energy has dropped precipitously from preceding frames while the spectral envelope has remained relatively constant. The average-frame-energy to current-frame-energy ratio (see Equation 2.4.4.3-7) and the differential LSP (see Equation 2.4.4.3-11) are used to make the temporally masked classification. Temporally masked frame detection is only enabled if the reduced rate level is 1, 2, or 3 and if the frame has not been classified as unvoiced (see Figure 2.4.4.3-1). Pseudocode defining the temporal masking classification is given as

```
{
  if(E_D > 15 and LSP_D < 0.02 and (RRM_Level == 2 or RRM_Level == 3)){
       mode = temporally masked
       rate = 1/4
  }
  else if(E_D > 9 and LSP_D < 0.02 and RRM_Level > 0 and RRM_Level < 4 ){
       mode = temporally masked
            rate = 1/2
  }
}
```

2.4.4.3.3 Stationary Voiced Frame Detection

If the reduced rate level, as defined in Table 2.3.5.2-2, is equal to three and if the frame has not been classified as unvoiced or temporally masked, then stationary voiced frames must be detected and encoded at Rate 1/2 to achieve the desired average rate for active speech (see Figure 2.4.4.3-1). The stationary voiced frame detection is based on Target_SNR, NACF, and PG_D (see 2.4.4.3), and can be done by the following pseudocode:

```
{
  if((Target_SNR > Target_SNR_Threshold) and (NACF > 0.4) and (PG_D > -5)){
        mode = stationary voiced
        rate = 1/2
  }
}
```

The logic behind this algorithm is as follows: if the previous frame was encoded accurately as measured by the Target_SNR, the current frame shows stationary periodicity as measured by the NACF function, and a sufficient level of prediction gain as measured by the differential prediction gain is maintained from the last frame, then the current frame is a good candidate frame for Rate 1/2 encoding (see Figure 2.4.4.3-1).

2.4.4.3.4 Adapting Thresholds to Achieve Target Average Rate

If the reduced rate level, as defined in Table 2.3.5.2-2, is equal to three, stationary voiced frames must be detected and encoded at Rate 1/2 to achieve the desired average rate for active speech. As defined in the pseudocode in 2.4.4.3.3, the number of Rate 1 frames that get encoded at Rate 1/2 will be a function of Target_SNR_Threshold. This threshold is adapted to maintain the average encoding rate for active speech close to the target value of 9.0 kbps. Note that this encoding rate is defined by the four channel rates of 14.4, 7.2, 3.6, and 1.8 kbps for Rate 1, Rate 1/2, Rate 1/4, and Rate 1/8, respectively.

The Target_SNR_Threshold is initialized to be 10 dB. After initialization, rate statistics are computed over analysis windows of 400 frames which are not Rate 1/8 packets, and Target_SNR_Threshold is adjusted appropriately. The average rate statistic computed over the active speech analysis window is calculated as

$$R_{AVG} = \frac{14.4(\#\ \text{Rate 1 frames}) + 7.2(\#\ \text{Rate 1/2 frames}) + 3.6\,(\#\ \text{Rate 1/4 frames})}{400} \qquad (2.4.4.3.4\text{-}1)$$

A histogram of the Target_SNR feature is also computed over the analysis window and is illustrated in Figure 2.4.4.3.4-1. This histogram counts the number of Rate 1 and Rate 1/2 frames with Target_SNR (see Equation 2.4.4.3-1) levels falling in 1 dB intervals in the range [Target_SNR_Threshold - 4, Target_SNR_Threshold + 3]. Thus, there are 8 bins in the histogram as shown in Figure 2.4.4.3.4-1. After the analysis time window has expired, the Target_SNR_Threshold is adjusted as follows:

```
{
  if(R_AVG > 1.02*9.0 )
     adjust Target_SNR_Threshold down
  if(R_AVG < 0.98*9.0)
     adjust Target_SNR_Threshold up
  else
     leave Target_SNR_Threshold unchanged
}
```

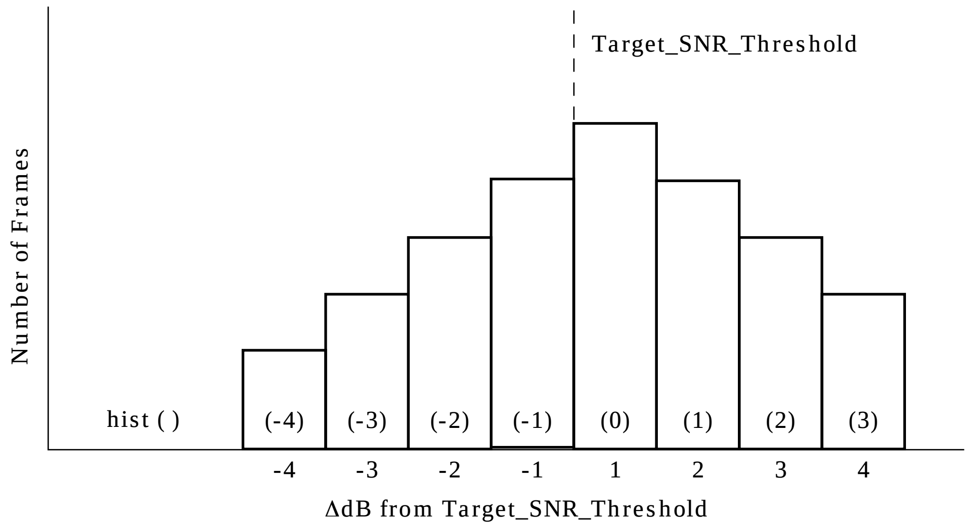


Figure 2.4.4.3.4-1. Histogram of Target_SNR Feature with Reference to Target_SNR_Threshold

The Target_SNR threshold is adjusted by moving the threshold in ±1 dB steps until enough Rate 1 frames (with Target_SNR measures below the threshold) would have been encoded at Rate 1/2 or enough Rate 1/2 frames (with Target_SNR measures above the threshold) would have been encoded at Rate 1 to meet the average rate target of 9.0. The Target_SNR_Threshold can only be moved by a maximum of ±4 dB in a single analysis window. The algorithm for adjusting the Target_SNR_Threshold is as follows:

```
{
  Ndelta1/2 = ((9.0-Ravg)/7.2)*400
  if(Ndelta1/2 > 0){ /* Rate is lower than the 9.0 kbps target */
     i = 0
     hist_total = 0
     while(hist_total < Ndelta1/2 && i < 4){
       hist_total += hist(i)
       i++
     }
     Target_SNR_Threshold = Target_SNR_Threshold +i
  }
  else{                     /* Rate is higher than the 9.0 kbps target */
     i = -1
     hist_total = 0
     while(hist_total < -Ndelta1/2 && i > -5){
       hist_total += hist(i)
       i--
     }
     Target_SNR_Threshold = Target_SNR_Threshold +i+1
  }
  if(Target_SNR_Threshold > 25)Target_SNR_Threshold = 25
  if(Target_SNR_Threshold < 6)Target_SNR_Threshold = 6
where
  hist(i) = # frames such that {T_S_T+i ≤ Target_SNR < T_S_T +i+1} for 0 ≤ i < 3
  hist(3) = # frames such that {T_S_T+3 ≤ Target_SNR}
  hist(i) = # frames such that {T_S_T+i+1 ≥ Target_SNR > T_S_T+i} for -3 ≤ i < 0
  hist(-4)= # frames such that {T_S_T-3 ≥ Target_SNR}
}
```

After each 400 frames of active speech the Target_SNR_Threshold is adapted to achieve the desired target rate of 9.0 kbps. At this time the histogram as shown in Figure 2.4.4.3.4-1 is reinitialized to zero. Thus, referring to the pseudocode above, hist(i) = 0 for $-4 \le i \le 3$.

The histogram for the Target_SNR feature is updated for every frame that is encoded at Rate 1 or Rate 1/2 according to the following pseudocode:

```
{
for(i=-4,i<3;i++){
  if(Target_SNR < Target_SNR_Threshold+i+1){
     hist(i) += 1;
     break;
  }
}
if(Target_SNR > Target_SNR_Threshold+3) hist(3)+=1;
}
```

2.4.5 Determining the Pitch Prediction Parameters

2.4.5.1 Encoding

All speech codec frames being encoded into Rate 1 or Rate 1/2 packets are subdivided into four pitch subframes, each of length 40 samples (see Table 2.4.1-1). There are no pitch subframes for Rate 1/8 or Rate 1/4 packets. The pitch synthesis filter can be expressed as

$$\frac{1}{P(z)} = \frac{1}{1 - bz^{-L}} \tag{2.4.5.1-1}$$

The pitch lag, L, is represented by 8 bits and ranges between 17 and 143 and includes fractional lags in units of 0.5 between 17 and 140. The pitch gain, b, is represented by three bits and ranges from 0 to 2.0 (see 2.4.5.1.3). For each pitch subframe, the speech codec determines and encodes the pitch lag, L, and the pitch gain b. The pitch lag, L, is selected from the set {17, 17.5,18,18.5,...,138.5,139,139.5,140,141,142,143} and the pitch gain, b, is selected from the set {0, 0.25, 0.5, . . . , 2.0}.

The method used to select the pitch parameters is an analysis-by-synthesis method, where encoding is done by selecting parameters which minimize the weighted error between the input speech and the synthesized speech using those parameters. The synthesized speech is the output of the formant synthesis (LPC) filter which processes the output of the pitch synthesis filter. The error between the input speech and the synthesized speech is weighted using the perceptual weighting filter

$$W(z) = \frac{A(z)}{A(z / \zeta)} \tag{2.4.5.1-2}$$

where A(z) is the formant prediction error filter and ζ, which is equal to 0.78, is a perceptual weighting parameter. The LPC coefficients $\hat{a}_i$ used in the perceptual weighting filter are those for the current pitch subframe (see 2.4.3.3.5 and 2.4.3.3.6).

Reduced processing can be obtained by the filter arrangement shown in Figure 2.4.5.1-1. See Table 2.4.5.1.1-1 for definitions of the symbols.

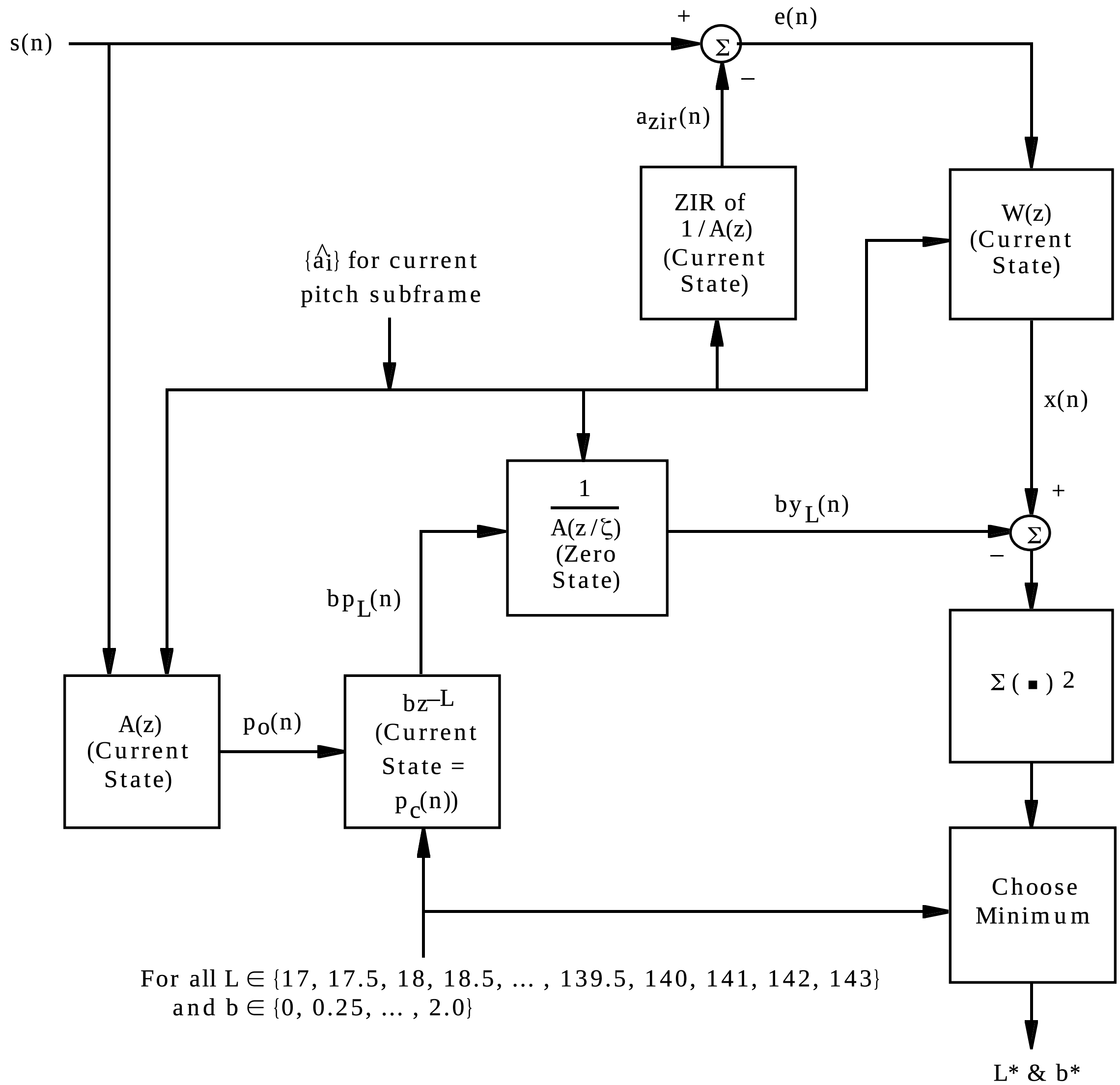


Figure 2.4.5.1-1. Analysis-by-Synthesis Procedure for the Pitch Parameter Search

In this form, the synthesis filter used in the speech encoder is called the weighted synthesis filter, which is the formant synthesis filter followed by the perceptual weighting filter, and is given by

$$H(z) = \left(\frac{1}{A(z)}\right)W(z) = \frac{1}{A(z/\xi)} \tag{2.4.5.1-3}$$

2.4.5.1.1 Computing the Pitch Lag and Pitch Gain

Table 2.4.5.1.1-1 lists the terms used to compute pitch lag and pitch gain.

Define

$$E_{xyL} = \sum_{n=0}^{L_p-1} x(n)y_L(n) \tag{2.4.5.1.1-1}$$

and

$$E_{yyL} = \sum_{n=0}^{L_p-1} {y_L}^2(n) \tag{2.4.5.1.1-2}$$

The optimal L, denoted by L*, and the optimal b, denoted by b*, are those values of L and b that result in the minimum value of

$$\sum_{n=0}^{L_p-1} \left\{x(n) - by_L(n)\right\}^2 \tag{2.4.5.1.1-3}$$

This minimum is computed by searching for the minimum of

$$-2bE_{xyL} + b^2E_{yyL} \tag{2.4.5.1.1-4}$$

over the allowable quantized values of L and b. The allowable quantized values are discussed in 2.4.5.1.

Table 2.4.5.1.1-1. Definition of Terms for Pitch Search

Term	Definition	Limits
L_P	Length, in samples, of the pitch subframe (see Table 2.4.1-1).	
s(n)	Input speech samples corresponding to the current pitch subframe with DC removed.	$0 \le n < L_P$
$\left\{\hat{a}_i\right\}$	LPC coefficients for the current pitch subframe.	
$a_{zir}(n)$	Zero input response, ZIR, of the formant synthesis filter, where 1/A(z) is initialized with the memories remaining in the decoder's 1/A(z) filter from the previous pitch subframe.	$0 \le n < L_P$
e(n)	$s(n) - a_{zir}(n)$	$0 \le n < L_P$
x(n)	e(n) filtered by W(z), where W(z) is initialized with the memories remaining in the decoder's W(z) filter after the last pitch subframe.	$0 \le n < L_P$
$p_c(n)$	Past outputs of the pitch synthesis filter. $p_c(-1)$ is the last output of the filter, $p_c(-2)$ is the second to last output, etc.	$-143 \le n < 0$
$p_o(n)$	An estimate of the future outputs of the pitch synthesis filter. This is s(n) filtered by A(z), using the appropriate LPC coefficients and states (previous input speech samples) for the current pitch subframe. This estimate is only used in the pitch search.	$0 \le n < L_P$
p(n)	Combined past outputs and estimated future outputs of the pitch synthesis filter, where $$p(n) = \begin{cases} p_c(n), & -143 \le n < 0 \\ p_o(n), & 0 \le n < L_P \end{cases}$$	$-143 \le n < L_P$
$p_L(n)$	p(n - L), the estimated output of the pitch synthesis filter for lag L, with b=1.	$0 \le n < L_P$
h(n)	Impulse response of H(z) truncated to length of N_{hp} elements for pitch search (See Equation 2.4.5.1-3)	$0 \le n < N_{hp}$
$y_L(n)$	$p_L(n)$ convolved with h(n).	$0 \le n < L_P$
L^*	Optimal pitch lag (see 2.4.5.1).	
b^*	Optimal pitch gain (see 2.4.5.1).	

2.4.5.1.2 Implementing the Pitch Search Convolutions

The zero state response of the weighted synthesis filter to $p_L(n)$, the estimated output of the pitch synthesis filter with lag L, can be calculated by convolving $p_L(n)$ with the impulse response of the weighted synthesis filter. The impulse response of the weighted synthesis filter H(z) can be truncated because it is typically small after 20 samples. With N_{hp} equal to 20, the convolution is approximated by

$$y_L(n) = \sum_{i=0}^{\min(n, N_{hp}-1)} h(i) p_L(n-i), \quad 16 < L \le 143 \text{ and } 0 \le n < L_P \qquad (2.4.5.1.2\text{-}1)$$

Note also that

$$p_L(n) = p(n-L) = p_{L-1}(n-1), \qquad 17 < L \le 143 \text{ and } 0 \le n < 40 \qquad (2.4.5.1.2\text{-}2)$$

From Equation 2.4.5.1.2-1 and Equation 2.4.5.1.2-2,

$$y_L(n) = \begin{cases} h(0)p(-L), & n = 0 \text{ and } 17 < L \le 143 \\ y_{L-1}(n-1) + h(n)p(-L), & 1 \le n < N_{hp} \text{ and } 17 < L \le 143 \\ y_{L-1}(n-1), & N_{hp} \le n < 40 \text{ and } 17 < L \le 143 \end{cases} \qquad (2.4.5.1.2\text{-}3)$$

In this way, once the initial convolution for $y_{17}(n)$ is computed using Equation 2.4.5.1.2-1, the remaining convolutions can be done recursively by Equation 2.4.5.1.2-3.

Fractional lags can be searched in a similar fashion to integer lags by upsampling the pitch memories, p(n), with an interpolation filter, such as described in Equation 2.4.5.2-2. Defining the 0.5 fractional pitch memories as

$$p_{L+0.5}(n) = p(n-(L+0.5)) = p_{L+0.5-1}(n-1), \; 17 < L \le 139 \text{ and } 0 \le n < 40 \qquad (2.4.5.1.2\text{-}4)$$

the fractional lag convolutions are approximated by

$$y_{L+0.5}(n) = \sum_{i=0}^{\min(n, N_{hp}-1)} h(i) p_{L+0.5}(n-i), \qquad 16 < L \le 139 \text{ and } 0 \le n < 40 \qquad (2.4.5.1.2\text{-}5)$$

From Equation 2.4.5.1.2-4 and Equation 2.4.5.1.2-5,

$$y_{L+0.5}(n) = \begin{cases} h(0)p(-(L+0.5)), & n = 0 \text{ and } 17 < L \le 139 \\ y_{L+0.5-1}(n-1) + h(n)p(-(L+0.5)), & 1 \le n < N_{hp} \text{ and } 17 < L \le 139 \\ y_{L+0.5-1}(n-1), & N_{hp} \le n < 40 \text{ and } 17 < L \le 139 \end{cases} \qquad (2.4.5.1.2\text{-}6)$$

In this way, once the initial fractional convolution for $y_{17.5}(n)$ is computed using Equation 2.4.5.1.2-5, the remaining convolutions can be done recursively by Equation 2.4.5.1.2-6.

2.4.5.1.3 Converting the Pitch Gain and Pitch Lag to the Transmission Codes

For each pitch subframe, the chosen parameters, b* and L*, are converted to transmission codes, PGAIN and PLAG. The chosen pitch gain, b*, which is a value from the set {0, 0.25,..., 2.0}, is linearly quantized between 0 and 2.0 in steps of 0.25. The chosen lag, L*, is an element in the set {17,17.5,18,...,138.5,139,139.5,140,141,142,143}.

The value of PLAG depends on both b* and L*. If b* = 0, then PLAG = 0. Otherwise, PLAG = L*-16-0.5 PFRAC. Thus, $\mathrm{PLAG} \in \{0,1,\ldots,127\}$ is represented using seven bits. The fractional value of the lag is coded using the bit PFRAC. If PFRAC equals zero the integer pitch is encoded. If PFRAC equals 1, then 0.5 is added to the integer pitch. The value of PGAIN depends only on b*. If b* = 0, then PGAIN = 0. Otherwise, PGAIN = b*/0.25 - 1. Thus, PGAIN is represented using three bits. Note that both b* = 0 and b* = 0.25 result in PGAIN = 0. These two cases are distinguished by the value of PLAG, which is zero in the first and non-zero in the second case. Fractional pitch lags 140.5,141.5,142.5, and 143.5 are invalid. If these invalid pitch lags are received the frame should be erased and processed as described in 2.4.8.7.1.

2.4.5.2 Decoding

To convert the transmission codes to pitch gain and pitch lag, the pitch parameters are decoded by the reverse of the transformation described in 2.4.5.1 (i.e., $\hat{b} = 0$ when PLAG = 0, otherwise $\hat{b} = (\mathrm{PGAIN} + 1)/4$ and $\hat{L} = \mathrm{PLAG} + 16 + 0.5\mathrm{PFRAC}$).

The pitch filter in the decoder is represented by the difference equation

$$p(n) = \hat{b}\, p\left(n - \hat{L}\right) + c_d(n) \tag{2.4.5.2-1}$$

where $c_d(n)$ is the scaled codebook vector. When PFRAC equals one ($\hat{L}$ is a fractional lag), $p\left(n - \hat{L}\right)$ is calculated using an 8th order interpolation filter

$$p\left(n - \hat{L}\right) = \sum_{i=-4}^{3} \mathrm{hammsinc}(i + 0.5)\, p\left(n + i - \left(\hat{L} - 0.5\right)\right) \tag{2.4.5.2-2}$$

where the hammsinc(·) function is defined as

$$\mathrm{hammsinc}(x) = \left(\frac{\sin \pi x}{\pi x}\right)\left(0.5 + 0.46\cos\left(\frac{\pi x}{4}\right)\right) \tag{2.4.5.2-3}$$

2.4.6 Determining the Excitation Codebook Parameters

2.4.6.1 Encoding

For Rate 1 and Rate 1/2 frames the speech codec determines the codebook index, I, and the codebook gain, G, for each codebook subframe. For Rate 1/4 and Rate 1/8 frames excitation codebooks are not searched, however the energy of the excitation signal is coded with a gain parameter that is obtained from the energy of the prediction residual (see 2.4.6.1.3). For Rate 1/4 frames, this excitation gain parameter is calculated five times per frame, while for Rate 1/8 frames the gain parameter is calculated once per frame.

The codebook parameters specify the excitation to the pitch filter. This excitation is formed by scaling a codebook vector by the codebook gain G. The goal of the codebook search is to determine the codebook vector and gain which minimize the weighed error between the input speech and the synthesized speech.

The two circular codebooks are given in Table 2.4.6.1-1 and Table 2.4.6.1-2. Each codebook consists of 128 values in signed decimal notation. The codebook in Table 2.4.6.1-1 is used for Rate 1/2 frames, and the one in Table 2.4.6.1-2 is used for Rate 1 frames. Note that the floating point values given in Tables 2.4.6.1-1 and 2.4.6.1-2 shall be quantized to fixed-point precision using

$$X_{int} = \frac{\left(\mathrm{round}\left(2^{13} X_{float}\right)\right)}{2^{13}} \qquad (2.4.6.1\text{-}1)$$

where X_{float} is the value from the table, round(x) is the function rounding to the closest integer, and X_{int} is the fixed-point precision number that shall be used in the algorithm implementation.

Table 2.4.6.1-1. Circular Codebook for Rate 1/2 Frames

n	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
c(n)	0.0	-2.0	0.0	-1.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

n	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
c(n)	0.0	-1.5	-1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.5

n	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
c(n)	0.0	0.0	0.0	0.0	0.0	0.0	2.0	0.0	0.0	1.5	1.0	0.0	1.5	2.0	0.0	0.0

n	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
c(n)	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.5	0.0	0.0

n	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
c(n)	-1.5	1.5	0.0	0.0	-1.0	0.0	1.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	-2.5	0.0

n	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
c(n)	0.0	0.0	0.0	1.5	0.0	0.0	0.0	1.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.0

n	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
c(n)	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.5	3.0	-1.5	-2.0	0.0	-1.5	-1.5

n	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
c(n)	1.5	-1.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Table 2.4.6.1-2. Circular Codebook for Rate 1 Frames

n	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
c(n)	0.10	-0.65	-0.59	0.12	1.10	0.34	-1.34	1.57	1.04	-0.84	-0.34	-1.15	0.23	-1.01	0.03	0.45

n	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
c(n)	-1.01	-0.16	-0.59	0.28	-0.45	1.34	-0.67	0.22	0.61	-0.29	2.26	-0.26	-0.55	-1.79	1.57	-0.51

n	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
c(n)	-2.20	-0.93	-0.37	0.60	1.18	0.74	-0.48	-0.95	-1.81	1.11	0.36	-0.52	-2.15	0.78	-1.12	0.39

n	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
c(n)	-0.17	-0.47	-2.23	0.19	0.12	-0.98	-1.42	1.30	0.54	-1.27	0.21	-0.12	0.39	-0.48	0.12	1.28

n	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
c(n)	0.06	-1.67	0.82	-1.02	-0.79	0.55	-0.44	0.48	-0.20	-0.53	0.08	-0.61	0.11	-0.70	-1.57	-1.68

n	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
c(n)	0.20	-0.56	-0.74	0.78	0.33	-0.63	-1.73	-0.02	-0.75	-0.53	-1.46	0.77	0.66	-0.29	0.09	-0.75

n	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
c(n)	0.65	1.19	-0.43	0.76	2.33	0.98	1.25	-1.56	-0.27	0.78	-0.09	1.70	1.76	1.43	-1.48	-0.07

n	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
c(n)	0.27	-1.36	0.05	0.27	0.18	1.39	2.04	0.07	-1.84	-1.97	0.52	-0.03	0.78	-1.89	0.08	-0.65

The method used to select the codebook vector and gain is an analysis-by-synthesis method similar to that used for the pitch parameters search procedure. The chosen codebook index, I*, and the chosen codebook gain, G*, are the allowable values of I and G which minimize the weighted error between the synthesized speech and the input speech. The synthesized speech is the scaled codebook vector, $c_d(n)$, filtered by the pitch synthesis filter and the formant synthesis (LPC) filter. The error between the input speech and the synthesized speech is weighted using the perceptual weighting filter defined in Equation 2.4.5.1-2.

Reduced processing can be obtained by the filter arrangement shown in Figure 2.4.6.1-1.

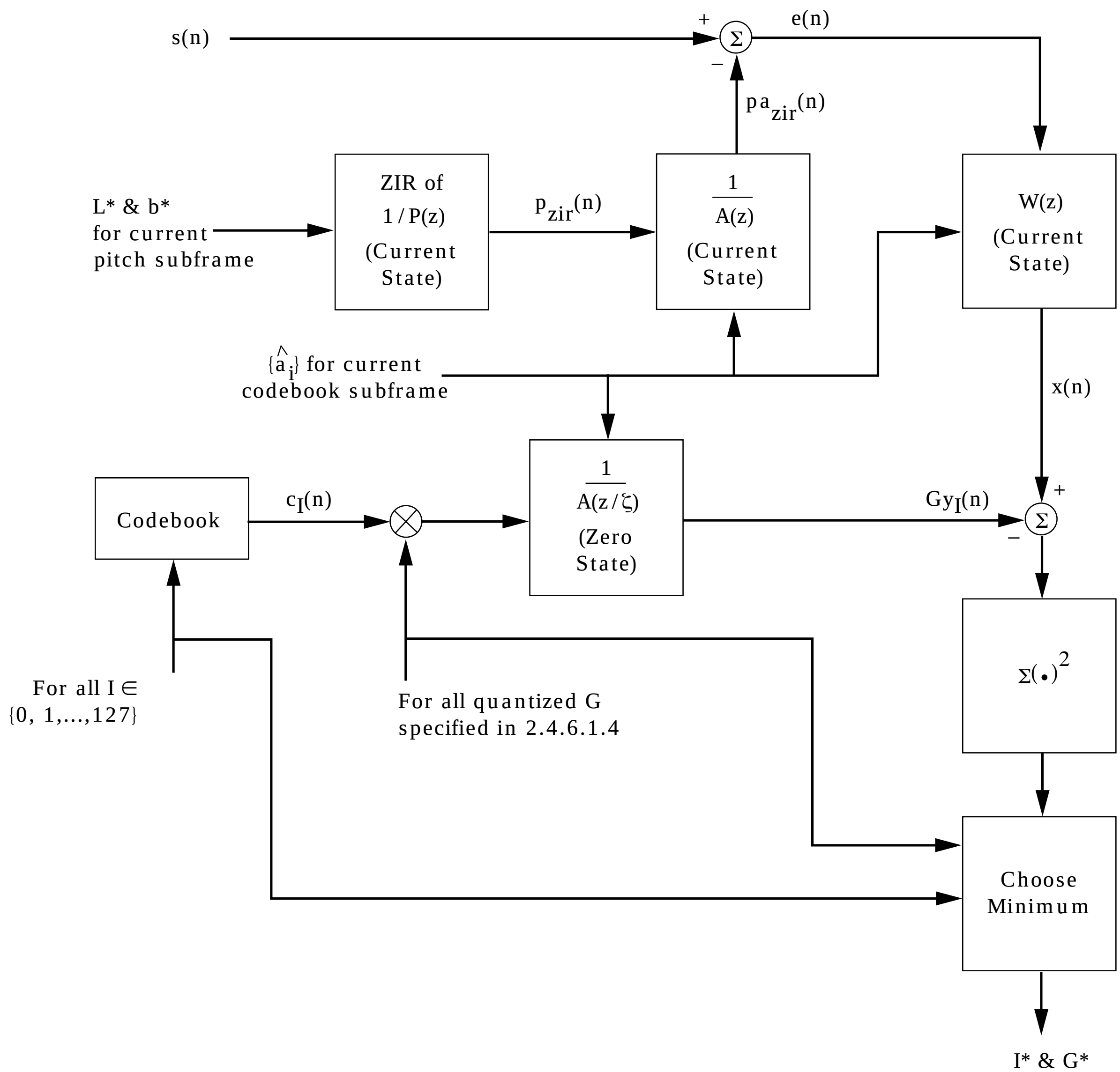


Figure 2.4.6.1-1. Analysis-by-Synthesis Procedure for Codebook Parameter Search

2.4.6.1.1 Computing the Codebook Index and Codebook Gain for Rate 1 and Rate 1/2

The following terms are used to compute codebook index and codebook gain.

Table 2.4.6.1.1-1. Definition of Terms for Codebook Search

Term	Definition	Limits
L_C	Length, in samples, of the codebook subframe (see Table 2.4.1-1).	
$s(n)$	Input speech samples corresponding to the current codebook subframe with DC removed.	$0 \le n < L_C$
$\left\{\hat{a}_i\right\}$	LPC coefficients for the current codebook subframe.	
$p_{zir}(n)$	Zero input response, ZIR, of the pitch synthesis filter, with L^* and b^* for the corresponding pitch subframe and $1/P(z)$ initialized with the memories remaining in the decoder's $1/P(z)$ filter after the last codebook subframe.	$0 \le n < L_C$
$pa_{zir}(n)$	$p_{zir}(n)$, filtered by $1/A(z)$, where $1/A(z)$ is initialized with the memories remaining in the decoder's $1/A(z)$ filter after the last codebook subframe.	$0 \le n < L_C$
$e(n)$	$s(n)$ - $pa_{zir}(n)$	$0 \le n < L_C$
$x(n)$	$e(n)$ filtered by $W(z)$, where $W(z)$ is initialized with the memories remaining in the decoder's $W(z)$ filter after the last codebook subframe.	$0 \le n < L_C$
$c(n)$	Circular codebook values.	$0 \le n < 128$
$c_I(n)$	The codebook vector for index I.	$0 \le n < L_C$
$h(n)$	Impulse response of $H(z)$ truncated to N_{hc} samples (see Equation 2.4.5.1-3)	$0 \le n < N_{hc}$
$y_I(n)$	$c_I(n)$ convolved with $h(n)$. This assumes that the impulse response of $1/P(z)$ is either simply an impulse over the entire codebook subframe length L_C, or that the pitch gain b is small, so that the effect of the impulse response of $1/P(z)$ is negligible. The pitch gain is typically only large at full rate when the codebook subframe size is sufficiently small, so the above assumption holds for all cases.	$0 \le n < L_C$
I^*	Index of the optimal codebook vector (see 2.4.6.1.1).	
G^*	Optimal codebook gain (see 2.4.6.1.1 and 2.4.6.1.3).	

Define

$$E_{xyI} = \sum_{n=0}^{L_C-1} x(n) y_I(n) \qquad (2.4.6.1.1\text{-}1)$$

and

$$E_{yyI} = \sum_{n=0}^{L_C-1} y_I^2(n) \tag{2.4.6.1.1-2}$$

The optimal I, denoted by I*, and the optimal G, denoted by G*, are those values of I and G that result in the minimum value of

$$\sum_{n=0}^{L_C-1} \left\{x(n) - Gy_I(n)\right\}^2 \tag{2.4.6.1.1-3}$$

This minimum is computed by searching for the minimum of

$$-2GE_{xyI} + G^2E_{yyI} \tag{2.4.6.1.1-4}$$

over the allowable quantized values of I and G. I may take any integer value from 0 to 127. The allowable quantized values of G are discussed in 2.4.6.1.4.

2.4.6.1.2 Implementing the Codebook Search Convolutions

Due to the recursive nature of the codebook, the same recursive convolution procedure used in the pitch search can be used in the codebook search. The zero state response of the weighted synthesis filter to $c_I(n)$, the codebook vector for index I, can be calculated by convolving $c_I(n)$ with the impulse response of the weighted synthesis filter. The impulse response of the weighted synthesis filter can be truncated because it is typically small after 20 samples. With N_{hc} equal to 20, the convolution is approximated by

$$y_I(n) = \sum_{i=0}^{\min(n,\, N_{hc}-1)} h(i)c_I(n-i), \qquad 0 \le I < 128 \text{ and } 0 \le n < L_C \tag{2.4.6.1.2-1}$$

The codebook vector for index I, $c_I(n)$, is defined as[8]

$$c_I(n) = \begin{cases} c\big((n-I)\bmod 128\big), & n-I \ge 0,\ 0 \le I < 128, \text{ and } 0 \le n < L_C \\ c\big(128 + (n-I)\big), & n-I < 0,\ 0 \le I < 128, \text{ and } 0 \le n < L_C \end{cases} \tag{2.4.6.1.2-2}$$

[8] For mod operations, see note 6 in the front matter.

From Equations 2.4.6.1.2-1 and 2.4.6.1.2-2,

$$y_I(n) = \begin{cases} h(0)c_I(0), & n = 0, \text{ and } 1 \le I < 128 \\ y_{I-1}(n-1) + h(n)c_I(0), & 1 \le n < N_{hc}, \text{ and } 1 \le I < 128 \\ y_{I-1}(n-1), & N_{hc} \le n < L_C, \text{ and } 1 \le I < 128 \end{cases} \qquad (2.4.6.1.2\text{-}3)$$

Once the initial convolution for $y_0(n)$ is completed using Equation 2.4.6.1.2-1, the remaining convolutions can be done recursively by Equation 2.4.6.1.2-3. When c((-I) mod 128) = 0, Equation 2.4.6.1.2-3 takes the simplified form

$$y_I(n) = \begin{cases} 0, & n = 0 \text{ and } 1 \le I < 128 \\ y_{I-1}(n-1), & 1 \le n < L_C \text{ and } 1 \le I < 128 \end{cases} \qquad (2.4.6.1.2\text{-}4)$$

2.4.6.1.3 Computing the Codebook Gain for Rate 1/4 and Rate 1/8 Frames

For Rate 1/4 and Rate 1/8 frames the codebook gain parameter is used to scale the pseudorandom excitation used for speech synthesis. This codebook gain parameter is obtained from the energy of the prediction residual. The energy of the prediction residual is calculated by scaling the input speech energy by the prediction gain ratio, $E^{(P)}/E^{(0)}$, as

$$Re(0) = \left(E^{(P)} / E^{(0)}\right) \sum_{n=0}^{L_f - 1} s^2(n) \qquad (2.4.6.1.3\text{-}1)$$

where L_f is the length of the subframe over which the codebook gain is being calculated. L_f equals 160 for Rate 1/8 frames and 32 for Rate 1/4 frames. The variables $E^{(P)}$ and $E^{(0)}$ are computed as described in the pseudocode in 2.4.3.2.4.

For Rate 1/8 frames, the codebook gain is then calculated using

$$G^* = \text{Suppression\_factor} \sqrt{\frac{Re(0)}{160}} \qquad (2.4.6.1.3\text{-}2)$$

where the Suppression_factor is calculated using the pseudocode

```
{
if(SNRf(1)(k) > 3 ){
  Suppression_factor = 0.3152
  hysteresis = 1
}
else if(SNRf(1)(k) < 2 ){
  Suppression_factor = 0.6304
  hysteresis = 0
}
else if (hysteresis == 1)
  Suppression_factor = 0.3152
else
  Suppression_factor = 0.6304
Note: hysteresis is set to 0 initially
}
```

$SNR_{f(1)}(k)$ is defined in Equation 2.4.4.1.2-3.

For Rate 1/4 frames, the codebook gain is calculated using

$$G^{*} = 1.2608\sqrt{\frac{Re(0)}{32}} \tag{2.4.6.1.3-3}$$

For Rate 1/4 frames, five codebook gains are computed per frame and Re(0) is computed over the appropriate 32-sample subframe.

2.4.6.1.4 Converting Codebook Parameters into Transmission Codes for Rate 1 and Rate 1/2

Figure 2.4.6.1.4-1 shows the conversion scheme used for Rate 1 and Rate 1/2 frames. Differential quantization of the codebook gain parameter is only used for every fourth codebook subframe during Rate 1 encoding. For all the other Rate 1 and Rate 1/2 codebook subframes, non-differential coding is used.

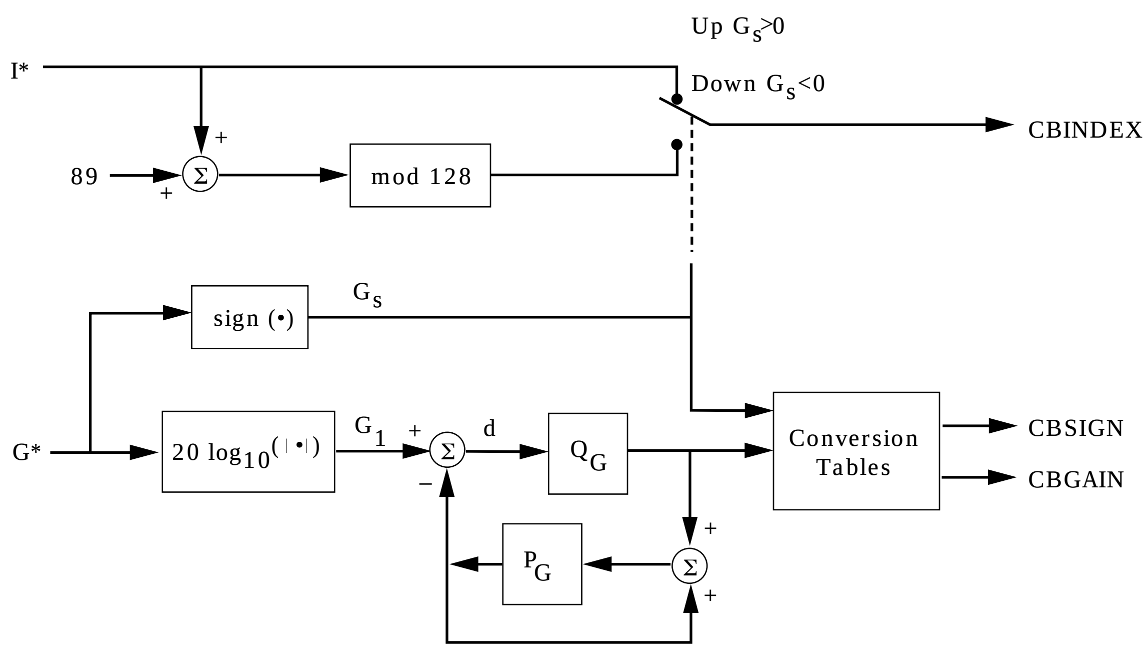


Figure 2.4.6.1.4-1. Converting Codebook Parameters for Rate 1 and Rate 1/2

For Rate 1 and Rate 1/2 frames, the sign, G_S, of the codebook gain, is

$$G_s = \mathrm{sign}\left(G^*\right) \tag{2.4.6.1.4-1}$$

where

$$\mathrm{sign}(x) = \begin{cases} 1, & x \geq 0 \\ -1, & x < 0 \end{cases} \tag{2.4.6.1.4-2}$$

The magnitude of the codebook gain is coded using a scalar quantizer operating on the log of the magnitude of G, as

$$G_1 = 20 \log_{10}\left(\left|G^*\right|\right) \tag{2.4.6.1.4-3}$$

The scalar quantizer employs either a 3- or 4-bit linear quantizer Q_G and a codebook gain predictor P_G, both of which depend on the encoding rate and subframe number. This quantizer operates once per codebook subframe. That is, the codebook gain is quantized 16 times during a Rate 1 frame and four times during a Rate 1/2 frame.

The predictor output, $P_G(x,n)$, at time n for an input sequence x(n) is

$$P_G(x, n) = \begin{cases} F_{G1}\left(\left\lfloor \dfrac{x(n-1) + x(n-2) + x(n-3)}{3} \right\rfloor\right), & \text{for every 4th subframe of Rate 1 frame} \\ 0.0, & \text{for other Rate 1 and all the Rate 1/2 subframes.} \end{cases} \tag{2.4.6.1.4-4}$$

where $\lfloor y \rfloor$ is the largest integer less than or equal to y, and $F_{G1}(y)$ is defined as

$$F_{G1}(y) = \begin{cases} y, & 6 < y < 38 \\ 6, & y \leq 6 \\ 38, & y \geq 38 \end{cases} \tag{2.4.6.1.4-5}$$

The input to the quantizer Q_G is formed as

$$d = G_1 - P_G(x, n) \tag{2.4.6.1.4-6}$$

The quantizer Q_G is shown in Tables 2.4.6.1.4-1 and 2.4.6.1.4-2.

Table 2.4.6.1.4-1. Codebook Quantizer (Rate 1, Rate 1/2, and Rate 1/4)

Range of d	$Q_G(d)$	Range of d	$Q_G(d)$
$d < 2$	0	$30 \le d < 34$	32
$2 \le d < 6$	4	$34 \le d < 38$	36
$6 \le d < 10$	8	$38 \le d < 42$	40
$10 \le d < 14$	12	$42 \le d < 46$	44
$14 \le d < 18$	16	$46 \le d < 50$	48
$18 \le d < 22$	20	$50 \le d < 54$	52
$22 \le d < 26$	24	$54 \le d < 58$	56
$26 \le d < 30$	28	$58 \le d$	60

Table 2.4.6.1.4-2. Codebook Quantizer (Rate 1 Every 4th Subframe)

Range of d	$Q_G(d)$
$d < -4$	-6
$-4 \le d < 0$	-2
$0 \le d < 4$	2
$4 \le d < 8$	6
$8 \le d < 12$	10
$12 \le d < 16$	14
$16 \le d < 20$	18
$20 \le d$	22

The output of the quantizer, $Q_G(d)$, and the sign, G_s, is converted to CBGAIN and CBSIGN, respectively, as shown in Tables 2.4.6.1.4-3 through 2.4.6.1.4-5.

Table 2.4.6.1.4-3. Conversion Table for CBGAIN (Rate 1, Rate 1/2, and Rate 1/4)

$Q_G(d)$	CBGAIN	$Q_G(d)$	CBGAIN
0	0	32	8
4	1	36	9
8	2	40	10
12	3	44	11
16	4	48	12
20	5	52	13
24	6	56	14
28	7	60	15

Table 2.4.6.1.4-4. Conversion Table for CBGAIN (Rate 1 Every 4th Subframe)

$Q_G(d)$	CBGAIN	$Q_G(d)$	CBGAIN
-6	0	10	4
-2	1	14	5
2	2	18	6
6	3	22	7

Table 2.4.6.1.4-5. Conversion Table for CBSIGN for Rate 1 and Rate 1/2

G_s	CBSIGN
+1	0
-1	1

If G_s is negative, CBINDEX is set equal to (I*+89) mod 128. If G_s is positive, CBINDEX is set equal to I*. This is done to reduce the sensitivity of the reconstructed speech signal to errors in the codebook gain sign bit.

2.4.6.1.5 Converting Codebook Parameters into Transmission Codes for Rate 1/4

The conversion scheme shown in Figure 2.4.6.1.5-1 is used only for Rate 1/4.

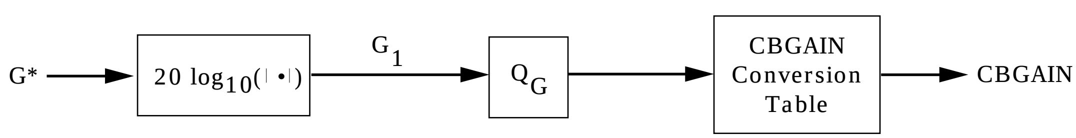


Figure 2.4.6.1.5-1. Converting Codebook Parameters for Rate 1/4

The magnitude of the codebook gain is coded using a scalar quantizer operating on the log of the magnitude of G, as

$$G_1 = 20 \log_{10}\left(\left|G^*\right|\right) \tag{2.4.6.1.5-1}$$

The scalar quantizer employs a 4-bit linear quantizer Q_G. This quantizer operates once per codebook subframe. That is, the codebook gain is quantized five times during a Rate 1/4 frame. G_l is quantized by Q_G as shown in Table 2.4.6.1.4-2, where d equals G_1. The output of the quantizer, $Q_G(d)$ is converted to CBGAIN as shown in Table 2.4.6.1.4-3.

For Rate 1/4 frames, the excitation codebook vector is replaced by a pseudorandom code vector in the decoding sections of the transmitting encoder and the receiving decoder. The codebook index and the sign of the codebook gain are not transmitted.

The pseudorandom code vector is generated by a pseudorandom number generator that is identical in the decoding sections of the transmitting encoder and the receiving decoder.

This is accomplished by using 16 bits from the data packet at Rate 1/4 as the seed for the pseudorandom number generator at both ends of transmission (see 2.4.8.1.2). These 16 bits from the Rate 1/4 packet are defined in Table 2.4.6.1.5-1 and are listed in order of MSB to LSB and the bit numbers are referenced to the packing table in 2.4.7.3-1.

Table 2.4.6.1.5-1. Rate 1/4 Frame Bits Used as the Seed for Pseudorandom Number Generation

Bit # in Rate 1/4 Packet (MSB to LSB)	Bit Description	Bit # in Rate 1/4 Packet (MSB to LSB)	Bit Description
33	LSPV5[1]	25	LSPV3[6]
32	LSPV5[0]	24	LSPV3[5]
31	LSPV4[5]	46	LSPV2[2]
30	LSPV4[4]	45	LSPV2[1]
29	LSPV4[3]	44	LSPV2[0]
28	LSPV4[2]	43	LSPV1[5]
27	LSPV4[1]	42	LSPV1[4]
26	LSPV4[0]	41	LSPV1[3]

2.4.6.1.6 Converting Codebook Parameters into Transmission Codes for Rate 1/8

The conversion scheme shown in Figure 2.4.6.1.6-1 is used only for Rate 1/8.

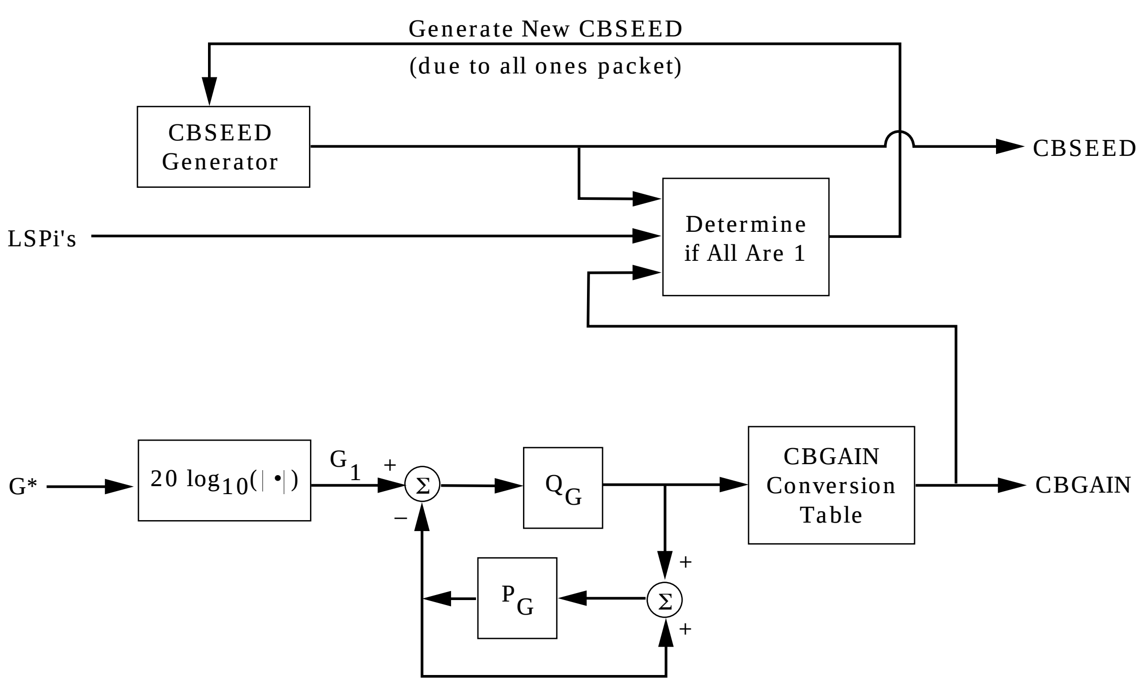


Figure 2.4.6.1.6-1. Converting Codebook Parameters for Rate 1/8

The magnitude of the codebook gain is coded using a scalar quantizer operating on the log of the magnitude of G*, as

$$G_1 = 20\ \log_{10}\left(\left|G^*\right|\right) \qquad (2.4.6.1.6\text{-}1)$$

The scalar quantizer employs a 2-bit linear quantizer Q_G and a codebook gain predictor P_G. The codebook gain is quantized once during a Rate 1/8 frame. The predictor output, $P_G(x,n)$, at time n for an input sequence x(n) is defined as

$$P_G(x,n) = F_{G2}\left(\mathrm{round}\left(\frac{x(n-1)+x(n-2)}{2}\right)\right) \qquad (2.4.6.1.6\text{-}2)$$

where round(y) is the function rounding to the closest integer, and $F_{G2}(y)$ is defined as

$$F_{G2}(y) = \begin{cases} y-1, & 4 < y < 59 \\ 4, & y \le 4 \\ 58, & y \ge 59 \end{cases} \qquad (2.4.6.1.6\text{-}3)$$

The input to the quantizer Q_G is formed as

$$d = G_1 - P_G(x,n)$$

The quantizer Q_G is shown in Table 2.4.6.1.6-1.

Table 2.4.6.1.6-1. Codebook Quantizer (Rate 1/8)

Range of d	Q_G(d)
d < -3	-4
-3 ≤ d < -1	-2
-1 ≤ d < 1	0
1 ≤ d	2

The output of the quantizer, Q_G(d), is converted to CBGAIN as shown in Table 2.4.6.1.6-2.

Table 2.4.6.1.6-2. Conversion Table for CBGAIN (Rate 1/8)

Q_G(d)	CBGAIN
-4	0
-2	1
0	2
2	3

For Rate 1/8 frames, the excitation codebook vector is replaced by a pseudorandom code vector in the decoding sections of the transmitting encoder and the receiving decoder. The codebook index and the sign of the codebook gain are not transmitted.

The pseudorandom code vector is generated by a pseudorandom number generator that is identical in the decoding sections of the transmitting encoder and the receiving decoder. This is accomplished by using the 16 non-reserved bits of the transmitted Rate 1/8 packet as the seed for the pseudorandom number generator at both ends of transmission (see 2.4.8.1.2).

CBSEED, which consists of four bits, is used to ensure the occurrence of random bit patterns in Rate 1/8 packets. These bits are generated by a pseudorandom number generator which generates relatively independent, uniformly distributed, pseudorandom numbers. A pseudorandom number generator using the integer SD_old which has been found to have satisfactory properties is

$$SD\_new = \left(521\left(SD\_old\right) + 259\right) \bmod 2^{16} \tag{2.4.6.1.6-4}$$

At transmitting encoder initialization, SD_old is set to 0.

For each new transmitted Rate 1/8 packet, SD_new is computed and the four bits of CBSEED are given by[9]

$$CBSEED\left[k\right] = SD\_new\left[4k + 3\right], \qquad k = 0,1,2,3 \tag{2.4.6.1.6-5}$$

where CBSEED[k] denotes bit k of CBSEED and SD_new [4k + 3] denotes bit 4k + 3 of the binary representation of SD_new. SD_old is set to SD_new for use in the next Rate 1/8 packet.

As an example, if SD_old = 40481 then

$$SD\_new = \left(521\left(40481\right) + 259\right) \bmod 2^{16} \tag{2.4.6.1.6-6}$$

In this case, CBSEED = ‘1001’, and SD_new = 53804 is saved for the next Rate 1/8 frame.

A Rate 1/8 packet with the format defined in 2.4.7.4 with all bits equal to 1 is equivalent to null Traffic Channel Data, as defined in EIA/TIA/IS-95, when the primary traffic field at the lowest negotiated transmission rate is 20 bits in length. If a packet with the first 16 bits equal to one occurs after packing (see 2.4.7.4), a new CBSEED is generated using the method above. The process is repeated until a CBSEED which is not all ones is generated. The packet is then repacked with the new CBSEED.

[9] See preface note 6 for an explanation of the bracket operator, [ ].

2.4.6.2 Decoding

2.4.6.2.1 Converting Codebook Transmission Codes for Rate 1 and Rate 1/2

Decoding of the codebook parameters is done by the reverse of the transformation described in 2.4.6.1.4. This is shown in Figure 2.4.6.2.1-1.

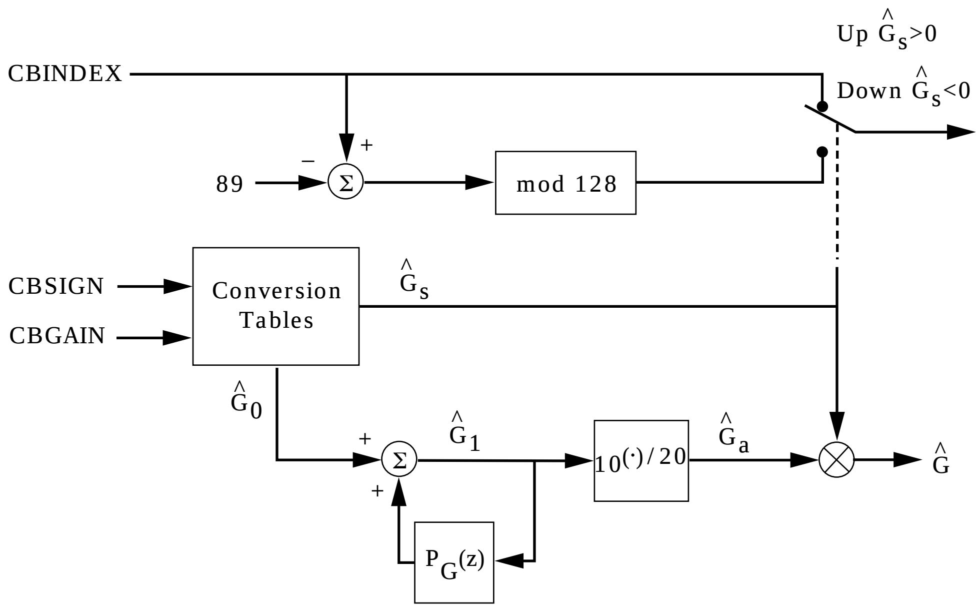


Figure 2.4.6.2.1-1. Converting Codebook Transmission Codes for Rate 1 and Rate 1/2

The encoding of the codebook transmission codes for Rate 1 and Rate 1/2 frames is done either differentially (for every fourth Rate 1 codebook subframe) or non-differentially (for the other Rate 1 subframes and all Rate 1/2 subframes). Therefore, the decoding of the codebook transmission codes is dependent upon the type of encoding. A codebook gain predictor, $P_G(z)$, is used to decode the differentially encoded codebook gain. For the non-differntially encoded codebook gain, $P_G(z)$ is equal to zero.

The codebook transmission parameter CBSIGN is converted to $\hat{G}_S$ using Table 2.4.6.2.1-1. CBGAIN is converted from the transmission code to $\hat{G}_0$ using Table 2.4.6.2.1-2. $\hat{G}_1$ is computed using $\hat{G}_0$ and the output, $P_G(x,n)$, of the codebook predictor where $P_G(x,n)$ at time n for an input sequence x(n) is defined in Equations 2.4.6.1.4-4 and 2.4.6.1.4-5.

The decoded $\hat{G}_1$ is converted back into the linear domain using Table 2.4.6.2.1-3. The values in this table correspond to the linear values of $\hat{G}_a$ with three fractional bits. Finally, $\hat{G}$ is calculated by multiplying $\hat{G}_a$ by $\hat{G}_S$.

If the received sign of the codebook gain $\hat{G}_S$ is equal to -1, the codebook index $\hat{I}$ is set to (CBINDEX - 89) mod 128. If $\hat{G}_S$ is equal to +1, $\hat{I}$ is set to CBINDEX.

Table 2.4.6.2.1-1. Table for Conversion from CBSIGN to $\hat{G}_S$

CBSIGN	$\hat{G}_S$
0	1
1	-1

Table 2.4.6.2.1-2. Table for Conversion from CBGAIN to $\hat{G}_0$

CBGAIN	$\hat{G}_0$
0	0
1	4
2	8
3	12
4	16
5	20
6	24
7	28
8	32
9	36
10	40
11	44
12	48
13	52
14	56
15	60

Table 2.4.6.2.1-3. Table for Conversion from $\hat{G}_1$ to $\hat{G}_a$

$\hat{G}_1$	$\hat{G}_a$
0	1.000
1	1.125
2	1.250
3	1.375
4	1.625
5	1.750
6	2.000
7	2.250
8	2.500
9	2.875
10	3.125
11	3.500
12	4.000
13	4.500
14	5.000
15	5.625
16	6.250
17	7.125
18	8.000
19	8.875
20	10.000
21	11.250
22	12.625
23	14.125
24	15.875
25	17.750
26	20.000
27	22.375
28	25.125
29	28.125
30	31.625
31	35.500
32	39.750
33	44.625
34	50.125
35	56.250
36	63.125
37	70.750
38	79.375
39	89.125
40	100.000
41	112.250
42	125.875
43	141.250
44	158.500
45	177.875
46	199.500
47	223.875
48	251.250
49	281.875
50	316.250
51	354.875
52	398.125
53	446.625
54	501.125
55	562.375
56	631.000
57	708.000
58	794.375
59	891.250
60	1000.000

2.4.6.2.2 Converting Codebook Transmission Codes for Rate 1/4

The procedure for determining the gain for Rate 1/4 is shown in Figure 2.4.6.2.2-1. For each of the five codebook gains the four bits of CBGAIN are converted to $\hat{G}_0$ using Table 2.4.6.2.1-2. The sign of the codebook gain, $\hat{G}_s$, is set to 1. The encoding for Rate 1/4 codebook gains is done non-differentially so $\hat{G}_1$ is equal to $\hat{G}_0$ and the unquantized codebook gains are converted to the linear domain using Table 2.4.6.2.1-3. Further, since $\hat{G}_S$ is equal to 1, $\hat{G}$ is equal to $\hat{G}_a$.

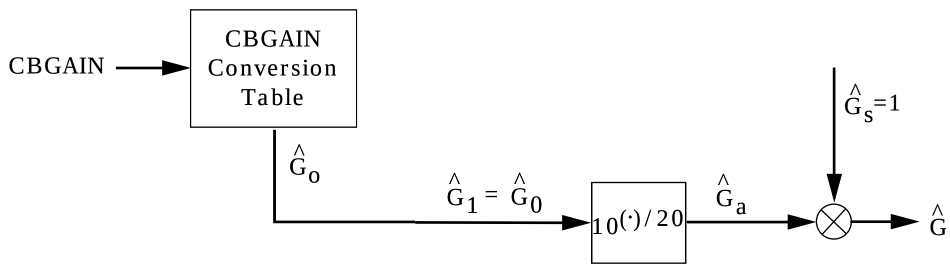


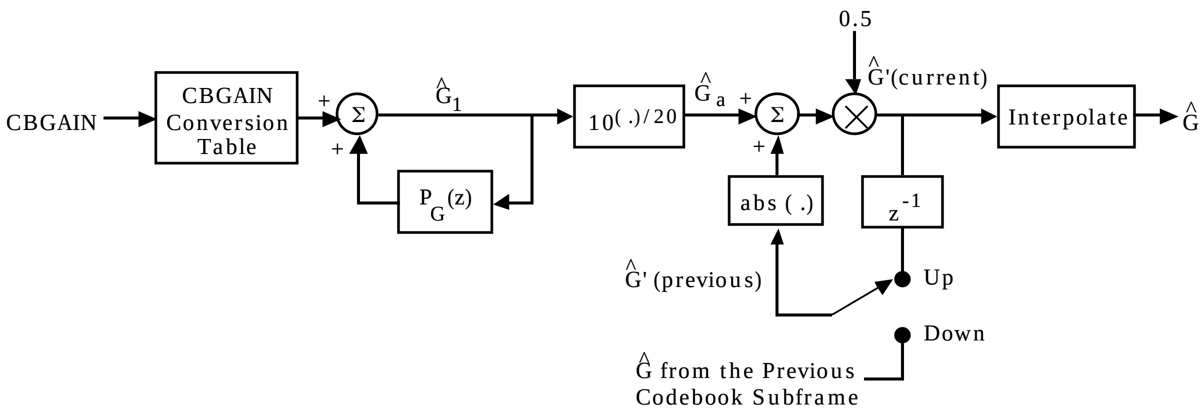
Figure 2.4.6.2.2-1. Converting Codebook Transmission Codes for Rate 1/4

To provide smoothing of the energy of the unvoiced excitation, the codebook gain is updated once per 20-sample segment of the 160-sample pseudorandom code vector. For each 20-sample segment, the five Rate 1/4 codebook gain parameters are used to generate the codebook gain via interpolation as shown by

$$\hat{G} = \begin{cases} \hat{G}(1), & 0 \le n < 20 \\ 0.6\,\hat{G}(1) + 0.4\,\hat{G}(2), & 20 \le n < 40 \\ \hat{G}(2), & 40 \le n < 60 \\ 0.2\,\hat{G}(2) + 0.8\,\hat{G}(3), & 60 \le n < 80 \\ 0.8\,\hat{G}(3) + 0.2\,\hat{G}(4), & 80 \le n < 100 \\ \hat{G}(4), & 100 \le n < 120 \\ 0.4\,\hat{G}(4) + 0.6\,\hat{G}(5), & 120 \le n < 140 \\ \hat{G}(5), & 140 \le n < 160 \end{cases} \qquad (2.4.6.2.2\text{-}1)$$

2.4.6.2.3 Converting Codebook Transmission Codes for Rate 1/8

The procedure for determining the gain for Rate 1/8 frames is shown in Figure 2.4.6.2.3-1. The least significant two bits of CBGAIN are converted back into -4, -2, 0, or 2 as shown in Table 2.4.6.1.6-2. The sign of the codebook gain, $\hat{G}_S$, is set to 1. The codebook index is not used in decoding Rate 1/8 packets (see 2.4.8.1.2).



Switch Up = Previous codebook subframe was from a Rate 1/8 packet
Switch Down = Previous codebook subframe was from other than a Rate 1/8 packet

Figure 2.4.6.2.3-1. Converting Codebook Transmission Codes for Rate 1/8

To prevent burstiness in the sound of the background noise, the current value of $\hat{G}_a$ is low-pass filtered as

$$\hat{G}'(\text{current}) = 0.5\left|\hat{G}'(\text{previous})\right| + 0.5\,\hat{G}_a(\text{current}) \qquad (2.4.6.2.3\text{-}1)$$

where $\hat{G}_a$ (current) is the decoded linear codebook gain for the current codebook frame, $\hat{G}'$ (previous) is the filtered linear codebook gain for the previous codebook frame or subframe, and $|x|$ is the absolute value of x. If the previous frame were at other than Rate 1/8, then $\hat{G}'$ (previous) is the codebook gain from the previous codebook subframe (e.g., $\hat{G}$ for the codebook subframe).

To produce smoother sounding background noise, the codebook gain is updated once per 20-sample segment of the 160-sample pseudorandom code vector, as is done for Rate 1/4 frames. For each 20-sample segment, the value of $\hat{G}'$ is used to generate the codebook gain via interpolation as shown by

$$\hat{G} = \begin{cases} 0.875\left|\hat{G}'(\text{previous})\right| + 0.125\,\hat{G}'(\text{current}), & 0 \le n < 20 \\ 0.750\left|\hat{G}'(\text{previous})\right| + 0.250\,\hat{G}'(\text{current}), & 20 \le n < 40 \\ 0.625\left|\hat{G}'(\text{previous})\right| + 0.375\,\hat{G}'(\text{current}), & 40 \le n < 60 \\ 0.500\left|\hat{G}'(\text{previous})\right| + 0.500\,\hat{G}'(\text{current}), & 60 \le n < 80 \\ 0.375\left|\hat{G}'(\text{previous})\right| + 0.625\,\hat{G}'(\text{current}), & 80 \le n < 100 \\ 0.250\left|\hat{G}'(\text{previous})\right| + 0.750\,\hat{G}'(\text{current}), & 100 \le n < 120 \\ 0.125\left|\hat{G}'(\text{previous})\right| + 0.875\,\hat{G}'(\text{current}), & 120 \le n < 140 \\ \hat{G}'(\text{current}) & 140 \le n < 160 \end{cases} \tag{2.4.6.2.3-2}$$

2.4.7 Data Packing

2.4.7.1 Rate 1 Packing

The 266 bits of a Rate 1 frame shall be packed into a primary traffic packet as shown in Table 2.4.7.1-1. Bit 266 shall be the first primary traffic bit in the frame and bit 0 shall be the last primary traffic bit in the frame. The reserved bits should be set to zero. If any one of these bits is received as a ‘1’, the received packet should be declared an insufficient frame quality (erasure) packet and the processing defined in 2.4.8.7.1 should be performed.

Table 2.4.7.1-1. Rate 1 Packet Structure (Part 1 of 3)

Bit	Code
265	LSPV3[2]
264	LSPV3[1]
263	LSPV3[0]
262	LSPV2[6]
261	LSPV2[5]
260	LSPV2[4]
259	LSPV2[3]
258	LSPV2[2]
257	LSPV2[1]
256	LSPV2[0]
255	LSPV1[5]
254	LSPV1[4]
253	LSPV1[3]
252	LSPV1[2]
251	LSPV1[1]
250	LSPV1[0]
249	LSPV5[5]
248	LSPV5[4]
247	LSPV5[3]
246	LSPV5[2]
245	LSPV5[1]
244	LSPV5[0]
243	LSPV4[5]
242	LSPV4[4]

Bit	Code
241	LSPV4[3]
240	LSPV4[2]
239	LSPV4[1]
238	LSPV4[0]
237	LSPV3[6]
236	LSPV3[5]
235	LSPV3[4]
234	LSPV3[3]
233	CBSIGN1[0]
232	CBGAIN1[3]
231	CBGAIN1[2]
230	CBGAIN1[1]
229	CBGAIN1[0]
228	PFRAC1[0]
227	PLAG1[6]
226	PLAG1[5]
225	PLAG1[4]
224	PLAG1[3]
223	PLAG1[2]
222	PLAG1[1]
221	PLAG1[0]
220	PGAIN1[2]
219	PGAIN1[1]
218	PGAIN1[0]

Bit	Code
217	CINDEX2[3]
216	CINDEX2[2]
215	CINDEX2[1]
214	CINDEX2[0]
213	CBSIGN2[0]
212	CBGAIN2[3]
211	CBGAIN2[2]
210	CBGAIN2[1]
209	CBGAIN2[0]
208	CINDEX1[6]
207	CINDEX1[5]
206	CINDEX1[4]
205	CINDEX1[3]
204	CINDEX1[2]
203	CINDEX1[1]
202	CINDEX1[0]
201	CBGAIN4[0]
200	CINDEX3[6]
199	CINDEX3[5]
198	CINDEX3[4]
197	CINDEX3[3]
196	CINDEX3[2]
195	CINDEX3[1]
194	CINDEX3[0]

Bit	Code
193	CBSIGN3[0]
192	CBGAIN3[3]
191	CBGAIN3[2]
190	CBGAIN3[1]
189	CBGAIN3[0]
188	CINDEX2[6]
187	CINDEX2[5]
186	CINDEX2[4]
185	PLAG2[2]
184	PLAG2[1]
183	PLAG2[0]
182	PGAIN2[2]
181	PGAIN2[1]
180	PGAIN2[0]
179	CINDEX4[6]
178	CINDEX4[5]
177	CINDEX4[4]
176	CINDEX4[3]
175	CINDEX4[2]
174	CINDEX4[1]
173	CINDEX4[0]
172	CBSIGN4[0]
171	CBGAIN4[2]
170	CBGAIN4[1]

Table 2.4.7.1-1. Rate 1 Packet Structure (Part 2 of 3)

Bit	Code
169	CINDEX5[5]
168	CINDEX5[4]
167	CINDEX5[3]
166	CINDEX5[2]
165	CINDEX5[1]
164	CINDEX5[0]
163	CBSIGN5[0]
162	CBGAIN5[3]
161	CBGAIN5[2]
160	CBGAIN5[1]
159	CBGAIN5[0]
158	PFRAC2[0]
157	PLAG2[6]
156	PLAG2[5]
155	PLAG2[4]
154	PLAG2[3]
153	CBGAIN7[2]
152	CBGAIN7[1]
151	CBGAIN7[0]
150	CINDEX6[6]
149	CINDEX6[5]
148	CINDEX6[4]
147	CINDEX6[3]
146	CINDEX6[2]
145	CINDEX6[1]
144	CINDEX6[0]
143	CBSIGN6[0]
142	CBGAIN6[3]
141	CBGAIN6[2]
140	CBGAIN6[1]
139	CBGAIN6[0]
138	CINDEX5[6]
137	CINDEX8[2]
136	CINDEX8[1]
135	CINDEX8[0]
134	CBSIGN8[0]
133	CBGAIN8[2]
132	CBGAIN8[1]
131	CBGAIN8[0]
130	CINDEX7[6]
129	CINDEX7[5]
128	CINDEX7[4]
127	CINDEX7[3]
126	CINDEX7[2]
125	CINDEX7[1]
124	CINDEX7[0]
123	CBSIGN7[0]
122	CBGAIN7[3]
121	CBGAIN9[0]
120	PFRAC3[0]
119	PLAG3[6]
118	PLAG3[5]
117	PLAG3[4]
116	PLAG3[3]
115	PLAG3[2]
114	PLAG3[1]
113	PLAG3[0]
112	PGAIN3[2]
111	PGAIN3[1]
110	PGAIN3[0]
109	CINDEX8[6]
108	CINDEX8[5]
107	CINDEX8[4]
106	CINDEX8[3]
105	CBSIGN10[0]
104	CBGAIN10[3]
103	CBGAIN10[2]
102	CBGAIN10[1]
101	CBGAIN10[0]
100	CINDEX9[6]
99	CINDEX9[5]
98	CINDEX9[4]
97	CINDEX9[3]
96	CINDEX9[2]
95	CINDEX9[1]
94	CINDEX9[0]
93	CBSIGN9[0]
92	CBGAIN9[3]
91	CBGAIN9[2]
90	CBGAIN9[1]
89	CINDEX11[3]
88	CINDEX11[2]
87	CINDEX11[1]
86	CINDEX11[0]

Table 2.4.7.1-1. Rate 1 Packet Structure (Part 3 of 3)

Bit	Code
85	CBSIGN11[0]
84	CBGAIN11[3]
83	CBGAIN11[2]
82	CBGAIN11[1]
81	CBGAIN11[0]
80	CINDEX10[6]
79	CINDEX10[5]
78	CINDEX10[4]
77	CINDEX10[3]
76	CINDEX10[2]
75	CINDEX10[1]
74	CINDEX10[0]
73	PGAIN4[1]
72	PGAIN4[0]
71	CINDEX12[6]
70	CINDEX12[5]
69	CINDEX12[4]
68	CINDEX12[3]
67	CINDEX12[2]
66	CINDEX12[1]
65	CINDEX12[0]
64	CBSIGN12[0]

Bit	Code
63	CBGAIN12[2]
62	CBGAIN12[1]
61	CBGAIN12[0]
60	CINDEX11[6]
59	CINDEX11[5]
58	CINDEX11[4]
57	CINDEX13[1]
56	CINDEX13[0]
55	CBSIGN13[0]
54	CBGAIN13[3]
53	CBGAIN13[2]
52	CBGAIN13[1]
51	CBGAIN13[0]
50	PFRAC4[0]
49	PLAG4[6]
48	PLAG4[5]
47	PLAG4[4]
46	PLAG4[3]
45	PLAG4[2]
44	PLAG4[1]
43	PLAG4[0]
42	PGAIN4[2]

Bit	Code
41	CINDEX14[5]
40	CINDEX14[4]
39	CINDEX14[3]
38	CINDEX14[2]
37	CINDEX14[1]
36	CINDEX14[0]
35	CBSIGN14[0]
34	CBGAIN14[3]
33	CBGAIN14[2]
32	CBGAIN14[1]
31	CBGAIN14[0]
30	CINDEX13[6]
29	CINDEX13[5]
28	CINDEX13[4]
27	CINDEX13[3]
26	CINDEX13[2]
25	CBGAIN16[2]
24	CBGAIN16[1]
23	CBGAIN16[0]
22	CINDEX15[6]
21	CINDEX15[5]
20	CINDEX15[4]

Bit	Code
19	CINDEX15[3]
18	CINDEX15[2]
17	CINDEX15[1]
16	CINDEX15[0]
15	CBSIGN15[0]
14	CBGAIN15[3]
13	CBGAIN15[2]
12	CBGAIN15[1]
11	CBGAIN15[0]
10	CINDEX14[6]
9	RESERVED
8	RESERVED
7	CINDEX16[6]
6	CINDEX16[5]
5	CINDEX16[4]
4	CINDEX16[3]
3	CINDEX16[2]
2	CINDEX16[1]
1	CINDEX16[0]
0	CBSIGN16[0]

2.4.7.2 Rate 1/2 Packing

The 124 bits of a Rate 1/2 frame shall be packed into a primary traffic packet as shown in Table 2.4.7.2-1. Bit 123 shall be the first primary traffic bit in the frame and bit 0 shall be the last.

Table 2.4.7.2-1. Rate 1/2 Packet Structure

Bit	Code	Bit	Code	Bit	Code	Bit	Code
123	LSPV3[2]	**91**	CBSIGN1[0]	**59**	PGAIN3[1]	**27**	PFRAC4[0]
122	LSPV3[1]	90	CBGAIN1[3]	58	PGAIN3[0]	26	PLAG4[6]
121	LSPV3[0]	89	CBGAIN1[2]	57	CINDEX2[6]	25	PLAG4[5]
120	LSPV2[6]	88	CBGAIN1[1]	56	CINDEX2[5]	24	PLAG4[4]
119	LSPV2[5]	87	CBGAIN1[0]	55	CINDEX2[4]	23	PLAG4[3]
118	LSPV2[4]	86	PFRAC1[0]	54	CINDEX2[3]	22	PLAG4[2]
117	LSPV2[3]	85	PLAG1[6]	53	CINDEX2[2]	21	PLAG4[1]
116	LSPV2[2]	84	PLAG1[5]	52	CINDEX2[1]	20	PLAG4[0]
115	LSPV2[1]	83	PLAG1[4]	51	CINDEX2[0]	19	PGAIN4[2]
114	LSPV2[0]	82	PLAG1[3]	50	CBSIGN2[0]	18	PGAIN4[1]
113	LSPV1[5]	81	PLAG1[2]	49	CBGAIN2[3]	17	PGAIN4[0]
112	LSPV1[4]	80	PLAG1[1]	48	CBGAIN2[2]	16	CINDEX3[6]
111	LSPV1[3]	79	PLAG1[0]	47	CBGAIN2[1]	15	CINDEX3[5]
110	LSPV1[2]	78	PGAIN1[2]	46	CBGAIN2[0]	14	CINDEX3[4]
109	LSPV1[1]	77	PGAIN1[1]	45	PFRAC2[0]	13	CINDEX3[3]
108	LSPV1[0]	76	PGAIN1[0]	44	PLAG2[6]	12	CINDEX3[2]
107	LSPV5[5]	**75**	PLAG2[5]	**43**	CINDEX3[1]	**11**	CINDEX4[6]
106	LSPV5[4]	74	PLAG2[4]	42	CINDEX3[0]	10	CINDEX4[5]
105	LSPV5[3]	73	PLAG2[3]	41	CBSIGN3[0]	9	CINDEX4[4]
104	LSPV5[2]	72	PLAG2[2]	40	CBGAIN3[3]	8	CINDEX4[3]
103	LSPV5[1]	71	PLAG2[1]	39	CBGAIN3[2]	7	CINDEX4[2]
102	LSPV5[0]	70	PLAG2[0]	38	CBGAIN3[1]	6	CINDEX4[1]
101	LSPV4[5]	69	PGAIN2[2]	37	CBGAIN3[0]	5	CINDEX4[0]
100	LSPV4[4]	68	PGAIN2[1]	36	PFRAC3[0]	4	CBSIGN4[0]
99	LSPV4[3]	67	PGAIN2[0]	35	PLAG3[6]	3	CBGAIN4[3]
98	LSPV4[2]	66	CINDEX1[6]	34	PLAG3[5]	2	CBGAIN4[2]
97	LSPV4[1]	65	CINDEX1[5]	33	PLAG3[4]	1	CBGAIN4[1]
96	LSPV4[0]	64	CINDEX1[4]	32	PLAG3[3]	0	CBGAIN4[0]
95	LSPV3[6]	63	CINDEX1[3]	31	PLAG3[2]		
94	LSPV3[5]	62	CINDEX1[2]	30	PLAG3[1]		
93	LSPV3[4]	61	CINDEX1[1]	29	PLAG3[0]		
92	LSPV3[3]	60	CINDEX1[0]	28	PGAIN3[2]		

2.4.7.3 Rate 1/4 Packing

The 54 bits of a Rate 1/4 frame shall be packed into a primary traffic packet as shown in Table 2.4.7.3-1. Bit 53 shall be the first primary traffic bit in the frame and bit 0 shall be the last primary traffic bit in the frame. The two reserved bits should be set to zero. If any one of these bits is received as a ‘1’, the received packet should be declared an insufficient frame quality (erasure) packet and the processing defined in 2.4.8.7.1 should be performed.

Table 2.4.7.3-1. Rate 1/4 Packet Structure

Bit	Code	Bit	Code
53	LSPV3[2]	26	LSPV4[0]
52	LSPV3[1]	25	LSPV3[6]
51	LSPV3[0]	24	LSPV3[5]
50	LSPV2[6]	23	LSPV3[4]
49	LSPV2[5]	22	LSPV3[3]
48	LSPV2[4]	**21**	CBGAIN4[3]
47	LSPV2[3]	20	CBGAIN4[2]
46	LSPV2[2]	19	CBGAIN4[1]
45	LSPV2[1]	18	CBGAIN4[0]
44	LSPV2[0]	17	CBGAIN3[3]
43	LSPV1[5]	16	CBGAIN3[2]
42	LSPV1[4]	15	CBGAIN3[1]
41	LSPV1[3]	14	CBGAIN3[0]
40	LSPV1[2]	13	CBGAIN2[3]
39	LSPV1[1]	12	CBGAIN2[2]
38	LSPV1[0]	11	CBGAIN2[1]
37	LSPV5[5]	10	CBGAIN2[0]
36	LSPV5[4]	9	CBGAIN1[3]
35	LSPV5[3]	8	CBGAIN1[2]
34	LSPV5[2]	7	CBGAIN1[1]
33	LSPV5[1]	6	CBGAIN1[0]
32	LSPV5[0]	**5**	RESERVED
31	LSPV4[5]	4	RESERVED
30	LSPV4[4]	3	CBGAIN5[3]
29	LSPV4[3]	2	CBGAIN5[2]
28	LSPV4[2]	1	CBGAIN5[1]
27	LSPV4[1]	0	CBGAIN5[0]

2.4.7.4 Rate 1/8 Packing

The 20 bits of a Rate 1/8 frame shall be packed into a primary traffic packet as shown in Table 2.4.7.4-1. Bit 19 shall be the first primary traffic bit in the frame and bit 0 shall be the last primary traffic bit in the frame. The four reserved bits should be set to zero. If any one of these bits is received as a '1', the received packet should be declared an insufficient frame quality (erasure) packet and the processing defined in 2.4.8.7.1 should be performed.

Table 2.4.7.4-1. Rate 1/8 Packet Structure

Bit	Code	Bit	Code	Bit	Code	Bit	Code
19	CBSEED[3]	14	LSP4[0]	9	LSP8[0]	4	CBGAIN1[0]
18	LSP1[0]	13	LSP5[0]	8	LSP9[0]	**3**	RESERVED
17	LSP2[0]	12	LSP6[0]	7	CBSEED[0]	2	RESERVED
16	LSP3[0]	11	CBSEED[1]	6	LSP10[0]	1	RESERVED
15	CBSEED[2]	10	LSP7[0]	5	CBGAIN1[1]	0	RESERVED

2.4.8 Decoding at the Transmitting Speech Codec and the Receiving Speech Codec

At the encoder on the transmit side, after each codebook subframe a version of the decoder shown in Figure 2.4.8-1 is run to update the filter states. At the receive side, the decoder shown in Figure 2.4.8-2 decodes the received parameters to produce $s_d(n)$, the reconstructed speech. The two decoders are quite similar.

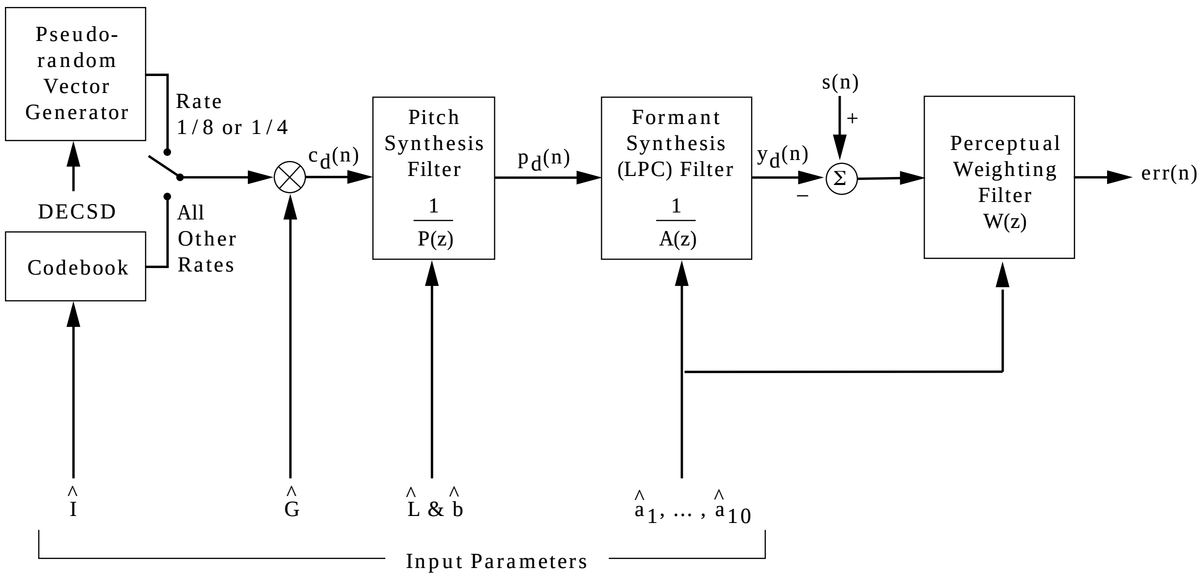


Figure 2.4.8-1. Decoding at the Transmitting Speech Codec

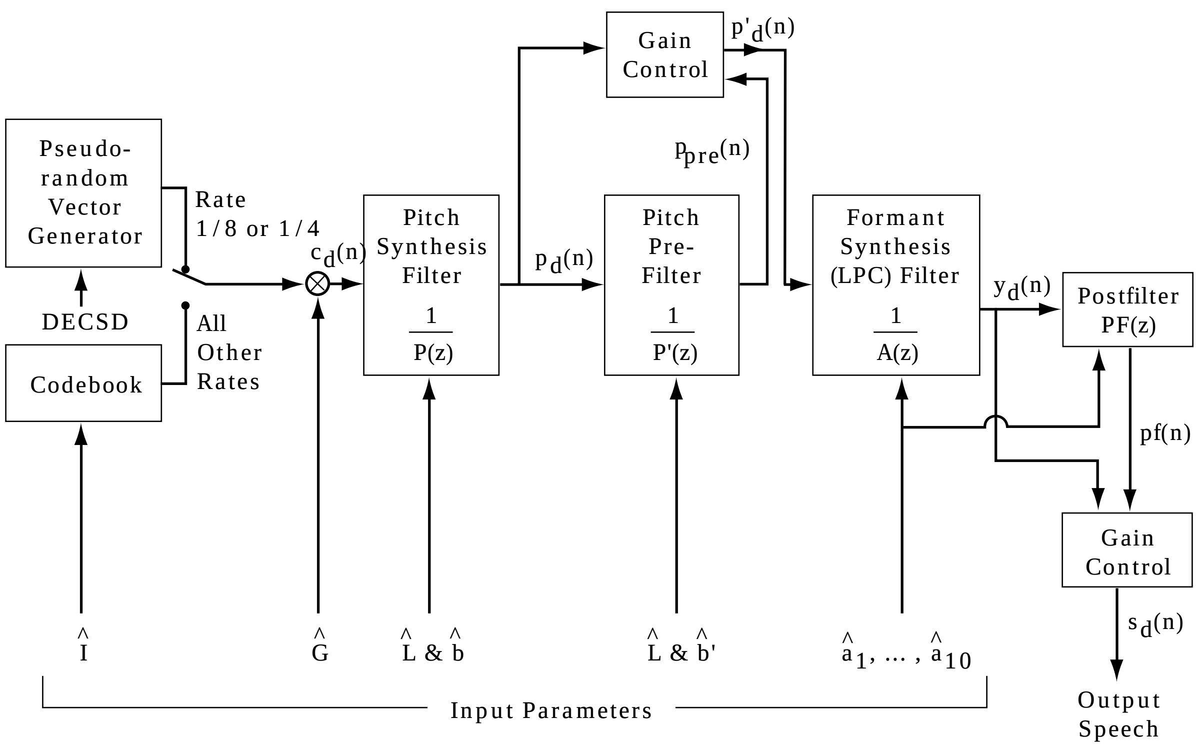


Figure 2.4.8-2. Decoding at the Receiving Speech Codec

2.4.8.1 Generating the Scaled Codebook Vector

Both the transmitting speech codec and the receiving speech codec generate the scaled codebook vector $c_d(n)$. $c_d(n)$ is generated differently for Rate 1/4 and Rate 1/8 packets than for all other rate packets.

2.4.8.1.1 Generating the Scaled Codebook Vector for Rate 1 and Rate 1/2

First, $\hat{I}$ and $\hat{G}$ are decoded from CBGAIN, CBINDEX and CBSIGN as described in 2.4.6.2.1. $c_d(n)$ is then set to $\hat{G}\,c\left(\left(n - \hat{I}\right) \bmod 128\right)$, where c(n) is the nth entry in the codebook shown in Table 2.4.6.1-1 or 2.4.6.1-2.

2.4.8.1.2 Generating the Scaled Codebook Vector for Rate 1/4

For Rate 1/4 frames, $c_d(n)$ is set to a pseudorandom white sequence. Both the transmitting speech codec and the receiving speech codec must produce exactly the same sequence. This requires that the pseudorandom number generators at both sides start with the exact same seed, hereafter referred to as DECSD.[10] DECSD is set to the 16-bit word derived from the Rate 1/4 packet as described in 2.4.6.1.5.

For generation of $c_d(n)$, a random sequence of length 160 is needed. This sequence, rnd(n), is generated using the following pseudocode.

```
{
  i=0
  decrv(old) = DECSD
  while (i < 160)
     {
       decrv(new) = (521*decrv(old) + 259) mod 2^16
       tmprnd = (decrv(new) + 2^15) mod 2^16 - 2^15
       rnd(i) = sqrt(1.887)*tmprnd /32768.0
       decrv(old) = decrv(new)
       i = i + 1
     }
  }
```

The temporary variable decrv is an integer, which is normalized to produce rnd(n) for each index n. The variable tmprnd is an integer.

rnd(n) is band-pass filtered before being scaled by the appropriate codebook gain. The filter states are saved from the last Rate 1/4 frame that used the filter. The sequence, rnd_bpf(n), is generated by band-pass filtering rnd(n) with the FIR filter described in Table 2.4.8.1.2-1.

---

[10] In an implementation which contains both the encoder and decoder operating in parallel, two distinct versions of DECSD must be kept so as not to confuse the pseudorandom number sequence generated at the encoder with the sequence generated at the decoder.

Table 2.4.8.1.2-1. Impulse Response of BPF Used to Filter the White Excitation for Rate 1/4 Synthesis

n	h(n)	n	h(n)
1	-1.344519E-1	12	3.749518E-2
2	1.735384E-2	13	-9.918777E-2
3	-6.905826E-2	14	3.501983E-2
4	2.434368E-2	15	-9.251384E-2
5	-8.210701E-2	16	3.041388E-2
6	3.041388E-2	17	-8.210701E-2
7	-9.251384E-2	18	2.434368E-2
8	3.501983E-2	19	-6.905826E-2
9	-9.918777E-2	20	1.735384E-2
10	3.749518E-2	21	-1.344519E-1
11	8.985137E-1		

The scaled code vector, $c_d(n)$, is determined using

$$c_d(n) = \hat{G}\ \mathrm{rnd\_bpf}(n) \tag{2.4.8.1.2-1}$$

where $\hat{G}$ is the interpolated gain value for the appropriate subframe (see 2.4.6.2.2). Although $c_d(n)$ is computed without fractional bits, rnd(n) and rnd_bpf(n) are computed using at least 12 bits of fractional precision, since the magnitude of rnd(n) is less than $\sqrt{1.887}$.

2.4.8.1.3 Generating the Scaled Codebook Vector for Rate 1/8

For Rate 1/8 frames, rnd(n) is set to a pseudorandom white sequence as was described for Rate 1/4 frames in 2.4.8.1.2. Both the transmitting speech codec and the receiving speech codec must produce exactly the same sequence. This requires that the pseudorandom number generators at both sides start with the exact same seed, hereafter referred to as DECSD. DECSD is set to the first 16 non-reserved bits of the 20-bit Rate 1/8 packet as described in 2.4.6.1.6.

The scaled code vector, $c_d(n)$, is determined using

$$c_d(n) = \hat{G}\,\mathrm{rnd}(n) \tag{2.4.8.1.3-1}$$

where $\hat{G}$ is the interpolated gain value for the appropriate subframe (see 2.4.6.2.3). Although $c_d(n)$ is computed without fractional bits, rnd(n) is computed with at least 12 bits of fractional precision, multiplied by the corresponding interpolated $\hat{G}$ value described above with three fractional bits, and then rounded to integer format.

2.4.8.2 Generating the Pitch Synthesis Filter Output

Both the transmitting speech codec and the receiving speech codec generate the output of the pitch synthesis filter, $p_d(n)$, identically. The filter $1/P(z)$ is initialized with the final state resulting from the last output sample generated, but using $\hat{b}$ and $\hat{L}$ appropriate for the current pitch subframe. $c_d(n)$ is filtered by $1/P(z)$ to produce $p_d(n)$. When $\hat{L}$ is non-integer, fractional pitch filtering is realized by interpolating the pitch memories as described in 2.4.5.2. For Rate 1/8 and Rate 1/4 frames, $\hat{b}$ is set to 0. The final state of the filter is saved for use in generating the output samples for the next pitch subframe and for use in the searches for the next pitch subframe in the encoder.

2.4.8.3 Generating the Pitch Pre-Filter Synthesis Output

The pitch pre-filter, $1/P'(z)$, resides only in the receiving speech codec. It is identical to $1/P(z)$ with the parameter $\hat{b}'$ replacing $\hat{b}$. The filter is initialized with the final state resulting from the last output sample generated, using $\hat{b}'$ and $\hat{L}$ appropriate for the current pitch subframe. $p_d(n)$ is filtered by $1/P'(z)$ to produce $p_{pre}(n)$. The pitch pre-filter gain coefficient, $\hat{b}'$, is derived from the pitch synthesis filter gain coefficient, $\hat{b}$, using

$$\hat{b}' = 0.5\ \min\left(\hat{b}, 1.0\right) \qquad (2.4.8.3\text{-}1)$$

Fractional pitch filtering is realized by interpolating the pitch memories as described in 2.4.5.2. For Rate 1/8 and Rate 1/4 packets, $\hat{b}'$ is set to 0. The final state of the filter is saved for use in generating the output samples for the next pitch subframe in the decoder.

A gain control should be put on the output of $1/P'(z)$ to ensure that the energy of the output signal is approximately the same as that of the input signal. The input and output energies are computed on 40-sample intervals only for Rate 1 and Rate 1/2 packets, since the pre-filter is effectively disabled for other rates. The gain control scale factor is computed as follows.

Compute the energy of $p_d(n)$ for each 40-sample subframe using

$$E_{inpre}(i) = \sum_{n=0}^{39} p_d^2(n + 40i), \qquad i = 0,1,2,3 \qquad (2.4.8.3\text{-}2)$$

where i=0 for the first subframe in the frame, i=1 for the second subframe in the frame, and so on.

Compute the energy of $p_{pre}(n)$ for each 40-sample subframe using

$$E_{outpre}(i) = \sum_{n=0}^{39} p_{pre}{}^2(n + 40i), \qquad i = 0,1,2,3 \qquad (2.4.8.3\text{-}3)$$

The gain control scale factor, GAIN(i), for the ith subframe is computed as

$$\mathrm{GAIN}(i) = \sqrt{\frac{E_{inpre}(i)}{E_{outpre}(i)}} \qquad (2.4.8.3\text{-}4)$$

The gain controlled pitch pre-filter output $p'_d(n)$ is generated using

$$p'_d(n + 40i) = \mathrm{GAIN}(i)\,p_{pre}(n + 40i), \quad i = 0,1,2,3 \qquad (2.4.8.3\text{-}5)$$

2.4.8.4 Generating the Formant Synthesis Filter Output

Both the transmitting speech codec and the receiving speech codec use identical formant synthesis filters. The LSP frequencies are interpolated as described in 2.4.3.3.4. The interpolated LSP frequencies are then converted back into LPC coefficients $\hat{a}_i$ as described in 2.4.3.3.5. The filter 1/A(z) is defined by these LPC coefficients and is initialized with the final state resulting from the last output sample generated. The filter output is denoted as $y_d(n)$. The final state of the filter is saved for use in generating future output samples, and for use in future pitch and codebook searches.

2.4.8.5 Updating the Memories of W(z) in the Transmitting Speech Codec

At the encoder, $s(n) - y_d(n)$ is filtered by W(z) (see 2.4.5.1) to update the filter memories of W(z) for use in future pitch and codebook searches. The filter W(z) is defined using the LPC coefficients $\hat{a}_i$ which are generated as described in 2.4.8.4 and is initialized with the final state resulting from the last output sample generated. The final state of the filter is saved for use in future searches.

2.4.8.6 The Adaptive Postfilter in the Receiving Speech Codec

At the decoder, an adaptive postfilter should be used to enhance the perceptual quality of the output speech. The postfilter has the form

$$\mathrm{PF}(z) = B(z)\frac{A(z/s)}{A(z/p)} \qquad (2.4.8.6\text{-}1)$$

where A(z) is the formant prediction error filter defined in Equation 2.4.3.1-1 but using the LPC coefficients $\hat{a}_i$ which are generated as described in 2.4.8.4, $s = 0.625$ and $p = 0.775$. B(z) is an anti-tilt filter designed to offset the spectral tilt introduced by A(z/s)/A(z/p). B(z) is given by

$$B(z) = \frac{1}{1 + 0.3z^{-1}} \qquad (2.4.8.6\text{-}2)$$

The filter PF(z) is initialized with the final state resulting from the last output sample. $y_d(n)$ should be filtered by PF(z) to produce pf(n).

A gain control should be put on the output of PF(z) to ensure that the energy of the output signal is approximately the same as the energy of the input signal. The input and output energies are computed on 40-sample subframes, regardless of the data rate selected. This is accomplished as follows:

Compute the energy of $y_d(n)$ for each 40-sample subframe using

$$E_{in}(i) = \sum_{n=0}^{39} y_d^{\,2}(n + 40i), \quad i = 0,1,2,3 \tag{2.4.8.6-3}$$

where i=0 for the first subframe in the frame, i=1 for the second subframe in the frame and so on.

Compute the energy of pf(n) for each 40-sample subframe using

$$E_{out}(i) = \sum_{n=0}^{39} pf^{\,2}(n + 40i), \qquad i = 0,1,2,3 \tag{2.4.8.6-4}$$

Compute tmpgain(i) using

$$tmpgain(i) = \sqrt{\frac{E_{in}(i)}{E_{out}(i)}}, \qquad i = 0,1,2,3 \tag{2.4.8.6-5}$$

This gain is filtered by a first order IIR filter to produce SCALE(i) using

$$SCALE(i) = 0.9375\,SCALE(i-1) + 0.0625\,tmpgain(i), \qquad i = 0,1,2,3 \tag{2.4.8.6-6}$$

where SCALE(-1) is equal to SCALE(3) from the previous frame.

Compute the reconstructed speech, $s_d(n)$, using

$$s_d(n + 40i) = SCALE(i)pf(n + 40i), \quad i = 0,1,2,3 \tag{2.4.8.6-7}$$

2.4.8.7 Special Cases

2.4.8.7.1 Insufficient Frame Quality (Erasure) Packets

If the received packet type cannot be satisfactorily determined, the multiplex sublayer informs the receiving speech codec of an erasure (see 2.3.2.2). In addition, the receiving speech codec declares an erasure in these three cases:

- When any of the reserved bits in the received packet are equal to 1 (see 2.4.7)
- When a Rate 1/8 packet consisting of all ones in the 16 non-reserved bit positions is received
- When an incorrect receive packet is detected by checking if the LSP frequencies or codebook gain parameters are outside normal bounds (see 2.4.8.7.3)

When the receiving speech codec receives or declares an erasure packet, the decoder decays some parameters toward their initialization levels. The current value of $\hat{G}_1$ is determined by subtracting the appropriate integer, N, from the previous value of $\hat{G}_1$ (the previous codebook gain in dB). The integer subtracted is a function of the number of consecutive erasures and is given in Table 2.4.8.7.1-1. If as a result of this subtraction $\hat{G}_1$ is less than 0 dB, $\hat{G}_1$ is set equal to 0 dB.

Table 2.4.8.7.1-1. Gain Subtraction Value as a Function of Consecutive Erasures

Number of Consecutive Erasures	N (in dB)
1	0
2	1
3	2
4 or more	6

$\hat{G}_1$ is entered into the predictor so that in the next frame, the output of the predictor will be a function of the average of the previous value of $\hat{G}_1$ in dB and the decremented gain used in the erasure frame. The linear codebook gain, $\hat{G}_a$, for the current frame is computed from the current value of $\hat{G}_1$ using Table 2.4.6.2.1-3. The current value of $\hat{G}_a$ is low-pass filtered as

$$\hat{G}'(\text{current}) = 0.5\left|\hat{G}'(\text{previous})\right| + 0.5\,\hat{G}_a(\text{current}) \qquad (2.4.8.7.1\text{-}1)$$

where $\hat{G}_a$ (current) is the decoded linear codebook gain for the current codebook frame, $\hat{G}'$ (previous) is the filtered linear codebook gain for the previous codebook frame or subframe, and $|x|$ is the absolute value of x. For each 40-sample segment, the value of $\hat{G}'$ is used to generate the codebook gain via interpolation as shown by

$$\hat{G} = \begin{cases} 0.750\left|\hat{G}'(\text{previous})\right| + 0.250\,\hat{G}'(\text{current}), & 0 \le n < 40 \\ 0.500\left|\hat{G}'(\text{previous})\right| + 0.500\,\hat{G}'(\text{current}), & 40 \le n < 80 \\ 0.250\left|\hat{G}'(\text{previous})\right| + 0.750\,\hat{G}'(\text{current}), & 80 \le n < 120 \\ \hat{G}'(\text{current}) & 120 \le n < 160 \end{cases} \qquad (2.4.8.7.1\text{-}2)$$

$\hat{G}_s$ is set equal to 1.

The codebook index, $\hat{I}$, is randomly chosen. The scaled codebook vector, $c_d(n)$, is generated using

$$c_d(n) = \hat{G}\, c\left(\left(n - \hat{I}\right) \bmod 128\right), \qquad 0 \le n < 160 \qquad (2.4.8.7.1\text{-}3)$$

where c(n) is the codebook in Table 2.4.6.1-2.

If the last frame received prior to an insufficient frame quality packet was Rate 1 or Rate 1/2, the following procedure is used to compute the pitch gain and lag. The pitch lag, $\hat{L}$, is repeated from the last pitch subframe of the previous frame. The pitch gain for the erased frame is saturated as

$$\hat{b} = \min\left(\text{previous subframes pitch gain}, b_e\right) \qquad (2.4.8.7.1\text{-}4)$$

where the pitch gain saturation value, b_e, is a function of the number of consecutive insufficient frame quality packets received as shown in Table 2.4.8.7.1-2.

Table 2.4.8.7.1-2. Pitch Saturation Levels as a Function of Consecutive Erasures

Number of Consecutive Erasures	Pitch Gain Saturation Value (b_e)
1	0.9
2	0.6
3	0.3
4 or more	0.0

If the last frame received prior to an insufficient frame quality was Rate 1/4 or Rate 1/8, the pitch lag and gain are set to zero for all consecutive erasure packets received.

The states in the LSP predictors are decayed by the predictor coefficient as a function of the number of consecutive erasure packets, where this function is defined in Table 2.4.8.7.1-3.

Table 2.4.8.7.1-3. LSP Predictor Decay as a Function of Consecutive Erasures

Number of Consecutive Erasures	LSP Predictor Coefficient
1	1.0
2	0.9
3	0.9
4 or more	0.7

The LSP frequencies are computed using the predictor and converted into LPC coefficients as in 2.4.3.3. The LPCs are bandwidth expanded to produce $\hat{a}_i$ and used for the entire frame of reconstructed speech.

$c_d(n), \hat{b}, \hat{L}$, and $\hat{a}_i$ are used to reconstruct the current frame of speech and to update the filter states for the next codebook subframe at the decoder.

2.4.8.7.2 Blank Packets

For a blank packet, the scaled codebook vector $c_d(n)$ is set equal to zero for the entire frame. The pitch lag, $\hat{L}$, is repeated from the last pitch subframe of the previous frame. The pitch gain, $\hat{b}$, is also repeated from the last pitch subframe of the previous frame with the exception that if the pitch gain is greater than 1, it is set equal to 1. The previous frame's uninterpolated LSP frequencies, $\hat{w}_i$, are converted into LPC coefficients. The LPCs are bandwidth expanded to reproduce $\hat{a}_i$ and used for the entire frame of reconstructed speech.

$c_d(n), \hat{b}, \hat{L}$, and $\hat{a}_i$ are used to reconstruct the current frame of speech and to update the filter states for the next codebook subframe at the decoder.

2.4.8.7.3 Incorrect Packet Detection

Before converting the LSP frequencies back to LPC coefficients for Rate 1, Rate 1/2, and Rate 1/4 frames, the LSP frequencies are checked to ensure that the resulting LPC filter is reasonable. An incorrect received packet, i.e., a Rate 1/8 packet received as a Rate 1/4 packet, can cause the LSPs to become too close together or can cause the LSPs to be greater than 1.0. If incorrect packets are detected, erasure processing should occur (see 2.4.8.7.1). The incorrect packet detection algorithm is described in the following pseudocode.

```
If rxrate == full or 1/2{
  if( .66>= wq(10 ) or wq(10)>=.985) erase packet
  for(n=5;n<11;n++)
     if(abs(wq(n)-wq(n-4)) < .0931) erase packet
}
If rxrate == 1/4{
  if( .70>= wq(10 ) or wq(10)>=.97 ) erase packet
  for(n=4;n<11;n++)
```

```
        if(abs(wq(n)-wq(n-3)) < .08) erase packet
    }
```

where wq(i) are the ten LSP frequencies (see 2.4.3.3.1) and rxrate is the encoded rate for the received packet.

If the received packet is Rate 1/4 a further sanity check is made of the codebook gain $\hat{G}_0$:

```
    If rxrate == 1/4{
      for(i = 0 ; i < 4; i++)
        if(abs( Ĝ0(i+1)- Ĝ0(i)) > 40) erase packet
      for(i = 0 ; i < 3; i++)
        if(abs( Ĝ0((i+2)-2 Ĝ0((i+1)+ Ĝ0((i)) > 48) erase packet
    }
```

where $\hat{G}_0(i)$ are the five Rate 1/4 codebook gain parameters represented in dB from 0 to 60 dB (see 2.4.6.2.2).

2.4.9 Initializing Speech Codec

Upon being commanded to initialize the receiving side, the speech codec sets all receiving parameters as follows:

- The filter and predictor memories are set to zero.
- The LSPs, $\hat{w}_i$ (previous frame), are set to $Bias_i$ (see 2.4.3.2.7 and 2.4.3.3.3).
- The Rate 1/8 codebook gain, $\hat{G}'$ (previous frame), is set to 0 (see 2.4.8.1.2).
- The adaptive postfilter gain, SCALE (previous), is set to 1.0 (see 2.4.8.6).
- The pitch gain and lag for the previous pitch subframe are set to zero (see 2.4.8.2).

Upon being commanded to initialize the transmitting side, the speech codec sets all transmitting parameters as follows:

- The filter and predictor memories are set to zero.
- The LSPs, $\hat{w}_i$ (previous), are set to $Bias_i$ (see 2.4.3.2.7 and 2.4.3.3.3).
- The Rate 1/8 codebook gain, $\hat{G}'$ (previous), is set to 0 (see 2.4.6.2.3).
- The smoothed signal energy estimate; $E^{sm}{}_{f(1)}(0)$ is set to 3200000 and $E^{sm}{}_{f(2)}(0)$ is set to 320000 (see 2.4.4.1).
- The background noise energy estimate $B_{f(i)}(0)$ is set to 5059644 for both frequency bands f(1) and f(2) (see 2.4.4.1.2)
- The signal energy estimate; $S_{f(1)}(0)$ is set to 3200000 and $S_{f(2)}(0)$ is set to 320000 (see 2.4.4.2.2).
- The feature parameters E_{AVG}, High Band $SNR_{lastframe}$, Low Band $SNR_{lastframe}$, Differential Prediction $Gain_{lastframe}$, Differential $LSP_{lastframe}$, and $(E_D)_{lastframe}$ are set to 0 (see 2.4.4.2).
- The Rate 1/8 random codebook seed, SD_old, is set to 0.

2.4.10 Output Audio Interface

2.4.10.1 Output Audio Interface in the Mobile Station

2.4.10.1.1 Band Pass Filtering

Output reconstruction filtering shall conform to CCITT Recommendation G.714 "Separate Performance Characteristics for the Encoding and Decoding Sides of PCM Channels Applicable to 4-Wire Voice-Frequency Interfaces." Additional reconstruction filtering may be provided by the manufacturer.

2.4.10.1.2 Adjusting the Receive Level

The mobile station shall have a nominal receive objective loudness rating (ROLR) equal to 51 dB when receiving from a reference base station (see 2.4.2.2.2). The loudness ratings are described in IEEE Standard 661-1979 "IEEE Standard Method for Determining Objective Loudness Ratings of Telephone Connections." Measurement techniques and tolerances are described in IS-125 "Recommended Minimum Performance Standard for Wideband Spread Spectrum Digital Cellular System Speech Service Options."

2.4.10.2 Output Audio Interface in the Base Station

Details of the digital and analog interfaces to the network are outside the scope of this document.

2.4.10.2.1 Adjusting the Receive Level

The base station shall set the audio level so that a received 1004 Hz tone 3.17 dB below maximum amplitude produces a level of 0 dBm0 at the network interface. Measurement techniques and tolerances are described in IS-125 "Recommended Minimum Performance Standard for Wideband Spread Spectrum Digital Cellular System Speech Service Options."

2.4.11 Summary of Encoding and Decoding

2.4.11.1 Encoding Summary

This section summarizes the steps taken to encode a frame:

1.0 Initial Computations

- 1.1 High-pass filter the current frame of input speech.
- 1.2 Compute the LPC coefficients for the current frame.
- 1.3 Compute the LSP frequencies from the LPC coefficients.
- 1.4 Perform the stage 1 rate determination.
- 1.5 Quantize the LSP frequencies and compute the LSP transmission codes.
- 1.6 Perform the stage 2 rate determination algorithm.
- 1.7 If the packet is Rate 1/2, go to 3.0.
- 1.8 If the packet is Rate 1/4, go to 4.0.

1.9 If the packet is Rate 1/8, go to 5.0.

1.10 If the packet is a Blank packet, go to 6.0.

1.11 Go to 2.0.

2.0 Rate 1 Packet Encoding

2.1 For the first pitch subframe in the frame,

2.2 Interpolate the LSPs for the pitch subframe and the four corresponding codebook subframes, and convert them to LPC coefficients.

2.3 Find the optimal pitch gain and lag for the pitch subframe,

2.4 For the first codebook subframe in the pitch subframe,

2.5 Find the optimal codebook gain and index.

2.6 Update the pitch synthesis filter, formant synthesis filter, and perceptual weighting filter states.

2.7 If all four codebook subframes in this pitch subframe frame have not been completed, go to the next codebook subframe and go to 2.5.

2.8 If all four pitch subframes for this frame have not been completed, go to the next pitch subframe and go to 2.2.

2.9 Pack the data into the 266-bit packet.

2.10 Done encoding.

3.0 Rate 1/2 Packet Encoding

3.1 For the first pitch subframe in the frame,

3.2 Interpolate the LSPs and convert them to LPC coefficients.

3.3 Find the optimal pitch gain and lag for the pitch subframe.

3.4 Find the optimal codebook gain and index for the codebook subframe.

3.5 Update the pitch synthesis filter, formant synthesis filter, and perceptual weighting filter states.

3.6 If all four pitch subframes for this frame have not been completed, go to the next pitch subframe and go to 3.2.

3.7 Pack the data into the 124-bit packet.

3.8 Done encoding.

4.0 Rate 1/4 Packet Encoding

4.1 Interpolate the LSP frequencies for four 40-sample subframes and convert them to LPC coefficients.

4.2 Compute the prediction residual for the entire frame using the interpolated LPC coefficients.

4.3 Compute the five codebook gains by calculating the RMS energy of the prediction residual in five 32-sample subframes.

4.4 Update the pitch synthesis filter, formant synthesis filter, and perceptual weighting filter states, using the DECSD.

4.5 Pack the data into the 54-bit packet.

4.6 Done encoding.

5.0 Rate 1/8 Packet Encoding

5.1 Interpolate the LSPs for the frame, then convert them to LPC coefficients.

5.2 Compute the prediction residual for the entire frame using the interpolated LPC coefficients.

5.3 Compute the codebook gain by calculating the RMS energy of the prediction residual for the entire frame.

5.4 Attenuate the codebook gain by the suppression factor.

5.5 Generate CBSEED, and pack the data into the 20-bit packet.

5.6 Update the pitch synthesis filter, formant synthesis filter, and weighting filter states.

5.7 Done encoding.

6.0 Blank Packet Encoding

6.1 Set the scaled codebook vector, $c_d(n)$ to zero.

6.2 Compute the pitch gain and lag.

6.3 Convert the previous frame's uninterpolated LSP frequencies to LPC coefficients.

6.4 Update the pitch, formant, and perceptual weighting filter states.

6.5 Done encoding.

2.4.11.2 Decoding Summary

The following summarizes the steps taken to decode a frame.

1.0 Initial Computations

1.1 If the received packet type is Rate 1/2, go to 3.0.

1.2 If the received packet type is Rate 1/4, go to 4.0.

1.3 If the received packet type is Rate 1/8, go to 5.0.

1.4 If the received packet type is blank, go to 7.0.

1.5 If the received packet is of insufficient frame quality (erasure), go to 6.0.

2.0 Rate 1 Packet Decoding

2.1 Unpack the 266-bit packet into transmission codes.

2.2 If any one of the reserved bits is not zero, go to 6.0.

2.3 Check for incorrectly received packets (see 2.4.8.7.3). If the packet is declared bad, go to 6.0.

2.4 Compute the speech codec parameters from the unpacked transmission codes.

2.5 Compute the scaled codebook vector for all 160 samples using the codebook index and gain parameters for all 16 codebook subframes.

2.6 Compute the output of the pitch synthesis filter for all 160 samples from the scaled codebook vector and the pitch lag and gain parameters for all four pitch subframes.

2.7 Compute the output of the pitch pre-filter for all 160 samples from the output of the pitch synthesis filter and the pitch lag and modified gain parameters for all four pitch subframes.

2.8 Interpolate the LSP frequencies for all four pitch subframes and convert these frequencies to LPC coefficients.

2.9 Compute the output of the formant synthesis filter for all 160 samples from the output of the pitch pre-filter and appropriate LPC coefficients for all four pitch subframes.

2.10 Compute output of the adaptive postfilter and the reconstructed speech for all 160 samples from the output of the formant synthesis filter and the LPC coefficients for all four pitch subframes.

2.11 Done decoding.

3.0 Rate 1/2 Packet Decoding

3.1 Unpack the 124-bit packet into transmission codes, and compute the speech codec parameters from these codes.

3.2 Check for incorrectly received packets (see 2.4.8.7.3). If the packet is declared bad, go to 6.0.

3.3 Compute the scaled codebook vector for all 160 samples using the codebook index and gain parameters for all four codebook subframes.

3.4 Compute output of the pitch synthesis filter for all 160 samples from the scaled codebook vector and the pitch lag and gain parameters for all four pitch subframes.

3.5 Compute the output of the pitch pre-filter for all 160 samples from the output of the pitch synthesis filter and the pitch lag and modified gain parameters for all four pitch subframes.

3.6 Interpolate the LSP frequencies for four pitch subframes and convert these frequencies to LPC coefficients.

3.7 Compute the output of the formant synthesis filter for all 160 samples from the output of the pitch pre-filter and appropriate LPC coefficients for all four pitch subframes.

3.8 Compute the output of the adaptive postfilter and the reconstructed speech for all 160 samples from the output of the formant synthesis filter and the LPC coefficients for all four pitch subframes.

3.9 Done decoding.

4.0 Rate 1/4 Packet Decoding

4.1 Unpack the 54-bit packet into the transmission codes, and compute the speech codec parameters from these codes.

4.2 If any one of the reserved bits is not zero, go to 6.0.

4.3 Check for incorrectly received packets (see 2.4.8.7.3). If the packet is declared bad, go to 6.0.

4.4 Compute the scaled codebook vector for all 160 samples using the designated 16-bit word from the packet as the random seed and the 5 codebook gain parameters.

4.5 Compute the output of the pitch synthesis filter for all 160 samples from the sealed codebook vector, with the pitch gain parameter set to zero.

4.6 Compute the output of the pitch pre-filter for all 160 samples from the output of the pitch synthesis filter with the pitch gain set to zero.

4.7 Interpolate the LSP frequencies for four subframes and convert these frequencies to LPC coefficients.

4.8 Compute the output of the formant synthesis filter for all 160 samples from the output of the pitch pre-filter and the LPC coefficients for four subframes.

4.9 Compute the output of adaptive postfilters and the reconstructed speech for all 160 samples from the output of the formant synthesis filter and the LPC coefficients for all four subframes.

4.10 Done decoding.

5.0 Rate 1/8 Packet Decoding

5.1 If the first 16 bits in the packet are all 1's, go to 6.0.

5.2 If any one of the reserved bits is not zero, go to 6.0.

5.2 Unpack the 20-bit packet into transmission codes and compute the speech codec parameters from these codes.

5.3 Compute the scaled codebook vector for all 160 samples using the first 16-bits in the packet as the random seed for the pseudorandom number generator and the codebook gain parameters.

5.4 Compute the output of the pitch synthesis filter for all 160 samples from the scaled codebook vector, with the pitch gain parameter set to zero.

5.5 Compute the output of the pitch pre-filter for all 160 samples from the output of the pitch synthesis filter with the pitch gain set to zero.

5.6 Interpolate the LSP frequencies and convert these frequencies to the LPC coefficients.

5.7 Compute the output of the formant synthesis filter for all 160 samples from the output of the pitch synthesis filter and the LPC coefficients.

5.8 Compute the output of the adaptive postfilter and the reconstructed speech for all 160 samples from the output of the formant synthesis filter and the LPC coefficients.

5.9 Done decoding.

6.0 Insufficient Frame Quality (Erasure) Decoding

6.1 Decay the codebook gain magnitude, update the codebook gain magnitude predictor states, and compute the linear value of the codebook gain.

6.2 Select a random codebook index.

6.3 Compute the scaled codebook vector for all 160 samples using the codebook gain and index parameters.

6.4 Compute the output of the pitch synthesis filter for all 160 samples from the scaled codebook vector and the smoothed pitch lag and gain parameters for all four pitch subframes.

6.5 Compute the output of the pitch pre-filter for all 160 samples from the output of the pitch synthesis filter and the smoothed pitch lag and gain parameters for all four pitch subframes.

6.6 Decay the LSP predictor states, compute the resulting LSP frequencies, and convert them into the LPC coefficients.

6.7 Compute the output of the formant synthesis filter for all 160 samples from the output of the pitch pre-filter and the LPC coefficients.

6.8 Compute the output of the adaptive postfilter and the reconstructed speech for all 160 samples from the output of the formant synthesis and the LPC coefficients.

6.9 Done decoding.

7.0 Blank Packet Decoding

7.1 Set the scaled codebook vector $c_d(n)$ to zero.

7.2 Compute the pitch gain and lag.

7.3 Compute the output of the pitch synthesis filter for all 160 samples using the pitch gain and lag.

7.4 Convert the previous frame's uninterpolated LSP frequencies to LPC coefficients.

7.5 Compute the output of the formant synthesis filter for all 160 samples from the output of the pitch synthesis filter and LPC coefficients.

7.6 Compute the output of the adaptive postfilter and the reconstructed speech for all 160 samples from the output of the formant synthesis filter and LPC coefficients.

7.7 Done decoding.

2.4.12 Allowable Delays

2.4.12.1 Allowable Transmitting Speech Codec Encoding Delay

The transmitting speech codec in the mobile station shall supply a packet to the multiplex sublayer not later than 20 ms after obtaining the last input sample for the Hamming window (see 2.4.3.2.2).

2.4.12.2 Allowable Receiving Speech Codec Decoding Delay

The receiving speech codec in the mobile station shall generate the first sample of speech using parameters from a packet received from the multiplex sublayer not later than 3 ms after receiving the packet.

2.5 Summary of Service Option 17 Notation

Table 2.5-1 lists the notation used by Service Option 17, Variable Data Rate Two-Way Voice.

1 **Table 2.5-1. Summary of Service Option 17 Notation (Part 1 of 6)**

Parameter	Section	Name/Description
$\alpha^{(P)}_{j}$	2.4.3.2.4	LPC coefficient j of formant synthesis (LPC) filter.
a_i	2.4.3.2.5	Linear predictive coding coefficients.
a'_i	2.4.3.3.5	Quantized, smoothed and interpolated LPC coefficients.
$\hat{a}_i$	2.4.3.3.6	Quantized, smoothed, interpolated, and bandwidth expanded LPC coefficients.
$a_{zir}(n)$	2.4.5.1.1	Zero input response of the formant synthesis filter.
A(z)	2.4.3.1	Formant prediction error filter.
1/A(z)	2.4.3.1	Formant synthesis filter.
b	2.4.5.1	Pitch gain.
b*	2.4.5.1.1	Optimal pitch gain.
b_e	2.4.8.7.1	Pitch gain saturation value used for erasure synthesis as derived from the previous pitch subframe.
$\hat{b}$	2.4.5.2	Pitch gain used for synthesis.
β	2.4.3.3.6	Scaling factor for bandwidth expansion.
$BE_{f(i)}$	2.4.4.1.1	Energy in the ith frequency band.
$B_{f(i)}(k)$	2.4.4.2.2	Background noise estimate for the ith frequency band in the kth frame.
$Bias_i$	2.4.3.2.7	Line spectral pair bias for LSP frequency i.
B(z)	2.4.8.6	Anti-tilt filter.
CBGAINi	2.4.1	Unsigned codebook gain for the ith codebook subframe.
CBINDEXi	2.4.1	Codebook index for the ith codebook subframe.
CBSEED	2.4.1	Four bit value to randomize Rate 1/8 packets.
CBSIGNi	2.4.1	Codebook gain sign for the ith codebook subframe.
$c_d(n)$	2.4.8.1	Scaled codebook vector.
$c_I(n)$	2.4.6.1.1	The codebook vector for index I.
c(n)	2.4.6.1.1	Circular codebook values.
d	2.4.6.1.4 2.4.6.1.5 2.4.6.1.6	Input to the quantizer Q_G.
decrv	2.4.8.1.2	Random variable used in generating the Rate 1/8 code vector.
DECSD	2.4.8.1.2 2.4.8.1.3	The decoder seed for Rate 1/4 and Rate 1/8 packets.
e(n)	2.4.5.1.1 2.4.6.1.1	The error between the input speech signal and the response of the formant synthesis filter.
$e_d(n)$	2.4.4.2.2	The signal obtaining after low-pass filtering e(n) and decimating by a factor of 2.
$E^{(i)}$	2.4.3.2.4	Energy of prediction error with formant synthesis (LPC) filter of order i.

Table 2.5-1. Summary of Service Option 17 Notation (Part 2 of 6)

Parameter	Section	Name/Description
$E^{sm}{}_{f(i)}(k)$	2.4.4.2.1	Smoothed energy estimate for the ith frequency band in the kth frame.
$E_{in}(i)$	2.4.8.6	Input energy to the adaptive postfilter.
$E_{inpre}(i)$	2.4.8.3	Input energy to the pitch pre-filter.
$E_D(k)$	2.4.4.3	The ratio of average-frame-energy to current-frame-energy.
$E_{out}(i)$	2.4.8.6	Output energy of the adaptive postfilter.
$E_{outpre}(i)$	2.4.8.3	Output energy of the pitch pre-filter.
E_{yyL}	2.4.5.1.1	The energy output of the weighted synthesis filter for the pitch search.
$F_{G1}(x)$	2.4.6.1.4	Codebook gain prediction filter function used for every fourth codebook subframe in Rate 1 frames.
$F_{G2}(x)$	2.4.6.1.6	Codebook gain prediction filter function used for Rate 1/8 frames.
f(i)	2.4.4.1.1	Frequency span of band-pass filter i.
G	2.4.6.1	Codebook gain.
G*	2.4.6.1	Optimal codebook gain.
$\hat{G}$	2.4.6.2.1	Decoded codebook gain.
$\hat{G}_a$	2.4.6.2.1	Decoded linear codebook gain magnitude.
$\hat{G}'$	2.4.6.2.3	Decoded and filtered codebook gain (used for Rate 1/8.
G_l	2.4.6.1.4	Codebook gain magnitude in dB.
$\hat{G}_l$	2.4.6.2.1	Decoded codebook gain magnitude in dB.
G_s	2.4.6.1.4	Sign of the codebook gain.
$\hat{G}_s$	2.4.6.2.1	Sign of the decoded codebook gain.
GAIN(i)	2.4.8.3	Gain control scale factor for the pitch pre-filter.
$h_i(n)$	2.4.4.1.1	Impulse response of the ith frequency band filter.
h(n)	2.4.5.1.2	Impulse response of H(z).
HPF(z)	2.4.3.2.1	High-pass filter used to pre-process the input speech signal before encoding begins.
H(z)	2.4.5.1	Weighted synthesis filter. The combined formant synthesis filter and perceptual weighting filter.
H_{SNR}	2.4.4.3	High Band Signal-to-Noise ratio estimate.
hammsinc(x)	2.4.5.2	Interpolation filter used to realize fractional pitch lags.

Table 2.5-1. Summary of Service Option 17 Notation (Part 3 of 6)

Parameter	Section	Name/Description
Hangover	2.4.4.1.4	Number of frames after a Rate 1 frame required before a non-Rate 1 frame can be encoded.
hist(i)	2.4.4.3.4	Histogram counters of the number of Rate 1 and Rate 1/2 frames having a Target_SNR above and below the Target_SNR_Threshold in 1 dB steps.
i	All sections	Index.
I	2.4.6.1	Codebook index.
I*	2.4.6.1	Index of optimal codeword.
$\hat{I}$	2.4.6.2.1	Codebook index used for synthesis.
k	All sections	Index.
λ	2.4.4.3	A leaky integrator used in average frame energy calculations
L	2.4.5.1	Pitch lag.
L*	2.4.5.1.1	Optimal pitch lag.
L_h	2.4.4.1.1	The length of the impulse response of the band-pass filters.
$\hat{L}$	2.4.5.2	Pitch lag used for synthesis.
L_A	2.4.1	LPC frame length in samples.
L_C	2.4.1	Codebook subframe length in samples.
L_P	2.4.1	Pitch subframe length in samples.
L_{SNR}	2.4.4.3	Low Band Signal-to-Noise ratio estimate.
$L_k(i,j)$	2.4.3.2.6.2	jth element of the kth vector in the ith LSP VQ codebook.
lownoise(i)	2.4.4.2.2	Lower bound on the background noise estimate in the ith frequency band.
LSPi	2.4.1	Transmission code for Line spectral pair frequency i.
ΔLSP(k)	2.4.4.3	Square of the magnitude of the difference between the last frames LSP vector and the current frames LSP vector.
LSP_D	2.4.4.3	ΔLSP(k) interpolated from the LPC frame to the encoding frame.
LSPVi	2.4.1	Transmission code for Line spectral pair frequency vector i.
N	2.4.3.2.7	Number of bits of quantization in $Q_w(x)$.
N_{hc}	2.4.6.1.2	Number of samples that are used from the impulse response of the weighted synthesis filter for codebook search.
N_{hp}	2.4.5.1.2	Number of samples that are used from the impulse response of the weighted synthesis filter for pitch search.
NACF	2.4.4.2.2	Normalized autocorrelation function.
P	2.4.3.1	Order of formant synthesis (LPC) filter.

Table 2.5-1. Summary of Service Option 17 Notation (Part 4 of 6)

Parameter	Section	Name/Description
$P_A(z)$	2.4.3.2.5	Intermediate polynomial used in transforming the LPC coefficients to LSP frequencies.
$\hat{P}_A(z)$	2.4.3.3.5	Intermediate polynomial used in transforming the interpolated LSP frequencies to LPC coefficients.
$pa_{zir}(n)$	2.4.6.1.1	Zero input response of the cascade of the pitch and formant synthesis filters.
$p_c(n)$	2.4.5.1.1	Past outputs of the pitch synthesis filter.
$p_d(n)$	2.4.8.2	Output of the pitch synthesis filter.
$p'_d(n)$	2.4.8.2	Output of the pitch pre-filter.
PF(z)	2.4.8.6	Adaptive post filter.
pf(n)	2.4.8.6	Output of the adaptive post filter.
PFRACi	2.4.1	Transmission code for the fractional part of the pitch lag for the ith pitch subframe.
P_G	2.4.6.1.4	Codebook gain predictor.
$P_G(x,n)$	2.4.6.1.4	Output of P_G at time n for input sequence x(n).
PGAINi	2.4.1	Transmission code for the pitch gain for the ith pitch subframe.
p_i	2.4.3.2.5	Coefficients of $P_A(z)$.
$\hat{p}_i$	2.4.3.3.5	Coefficients of $\hat{P}_A(z)$.
p'_i	2.4.3.2.5	Coefficients of P'(w).
$p_L(n)$	2.4.5.1.1	Estimated output of the pitch synthesis filter for lag L with b = 1.
PLAGi	2.4.1	Pitch lag for the ith pitch subframe.
p(n)	2.4.5.1.1	Combined past outputs and estimated future outputs of the pitch synthesis filter.
$p_o(n)$	2.4.5.1.1	Estimate of the future outputs of the pitch synthesis filter.
P'(w)	2.4.3.2.5	Function used in computing LSP frequencies.
$P_g(k)$	2.4.4.3	Energy in prediction gain expressed in dB.
$\Delta P_g(k)$	2.4.4.3	Differential prediction gain.
PG_D	2.4.4.3	Differential prediction gain interpolated from the LPC frame to the encoding frame.
$P_{w_i}(z)$	2.4.3.2.7	Prediction filter used in converting LSP frequencies.
1/P(z)	2.4.1	Pitch synthesis filter.
$p_{zir}(n)$	2.4.6.1.1	Zero input response of the pitch synthesis filter.
$Q_A(z)$	2.4.3.2.5	Intermediate polynomial used in transforming the LPC coefficients to LSP frequencies.
$\hat{Q}_A(z)$	2.4.3.3.5	Intermediate polynomial used in transforming the interpolated LSP frequencies to LPC coefficients.
$Q_G(x)$	2.4.6.1.4	Codebook gain quantizer function.

Table 2.5-1. Summary of Service Option 17 Notation (Part 5 of 6)

Parameter	Section	Name/Description
q_i	2.4.3.2.5	Coefficients of $Q_A(z)$.
$\hat{q}_i$	2.4.3.3.5	Coefficients of $\hat{Q}_A(z)$.
q'_i	2.4.3.2.5	Coefficients in $Q'(w)$.
$Q'(w)$	2.4.3.2.5	Function used in computing LSP frequencies.
$Q_w(x)$	2.4.3.2.7	Quantizer for LSP frequencies.
$R(k)$	2.4.3.2.3	kth value of the autocorrelation function for the current frame.
Re(0)	2.4.6.1.3	Subframe energy in the prediction residual used to derive the gain parameters for Rate 1/4 and Rate 1/8 frames.
R_{AVG}	2.4.4.3.4	The average rate statistic computed over the 8-second active speech analysis window used when the reduced rate level is equal to three.
R_D	2.4.4.3	Interpolated frame energy used as an input in the average frame energy filter
$R_{f(i)}(k)$	2.4.4.1.1	Autocorrelation function of the ith frequency band impulse response.
SD_old	2.4.6.1.6	Random number used to generate CBSEED in a Rate 1/8 packet.
SD_new	2.4.6.1.6	Temporary variable.
$S_{f(i)}(k)$	2.4.4.2.3	Signal energy estimate in the ith frequency band for the kth frame.
$S_w(n)$	2.4.3.2.2	Windowed input speech signal.
SCALE(i)	2.4.8.6	Scale factor for the adaptive postfilter in the receiving speech codec.
SM	2.4.3.3.3	Low-pass filter coefficient for the LSP frequency low-pass filter.
$SNR_{f(i)}(k)$	2.4.4.1.2	Quantized Signal-to-Noise Ratio in the ith frequency band for the kth frame.
SW_i	2.4.3.2.6.1	Sensitivity weighting for the ith LSP frequency used in VQ distortion measure.
$s(n)$	2.4.3.2.2	Input speech samples corresponding to the frame or subframe with DC removed.
$s_d(n)$	2.4.8	Speech reconstructed by the receiving speech codec.
$T_j(B,SNR)$	2.4.4.1.2	Thresholds used to determine the data rate as a function of the background noise and the quantized SNR in each frequency band i.
Target_SNR	2.4.4.3	Signal-to-Noise ratio between the target signal, $x(n)$, and the synthesized speech signal at the encoder, $y(n)$.
w_i	2.4.3.2.5	LSP frequencies.
$\hat{w}_i$	2.4.3.3.3	wq_i after stabilization and filtering.
$\hat{w}'_i$	2.4.3.3.4	$\hat{w}_i$ after interpolation.

1 **Table 2.5-1. Summary of Service Option 17 Notation (Part 6 of 6)**

Parameter	Section	Name/Description
wq_{min}	2.4.3.3.2	Minimum LSP frequency spacing.
wq_i	2.4.3.2.6.2	Quantized LSP frequencies
Δw_i	2.4.3.2.6.2	w_i-$w_{(i-1)}$
$W_H(n)$	2.4.3.2.2	Hamming window.
W(z)	2.4.5.1	Perceptual weighting filter.
x(n)	2.4.5.1.1	e(n) filtered by W(z).
$y_d(n)$	2.4.8.4	Formant synthesis filter output.
$y_I(n)$	2.4.6.1.1	$c_I(n)$ convolved by h(n).
$y_L(n)$	2.4.5.1.1	$p_L(n)$ convolved by h(n).
z	All sections	z transform variable.
ζ	2.4.5.1	Perceptual weighting parameter used in W(z).

No text.

ANNEX A BIBLIOGRAPHY

This is an informative annex. The documents listed in this annex are for information only and are not essential for the completion of the requirements of this standard.

—Other Standards

1. TIA/EIA/IS-96-B, *Speech Service Option Standard for Wideband Spread Spectrum System*, July 1996

—Books:

1. Rabiner, L. R. and Schafer, R. W., Digital Processing of Speech Signals, New Jersey, Prentice-Hall Inc., 1978.

2. Oppenheim, A. V. and Schafer, R. W., Digital Signal Processing, New Jersey, Prentice-Hall, Inc., 1975.

No text.