APPENDIX A REQUIREMENTS FOR CDMA SERVICE OPTIONS

This appendix describes the service options that can be used with the CDMA system. Service options are user application services (see Appendix C) provided by the system.

Service options are referenced via 16-bit numbers. Service option numbers from 1 through 32767 are reserved for definition by this specification and shall not be defined by manufacturers. Service option numbers from 32768 through 49151 may be defined by mobile station manufacturers. Service option numbers from 49152 through 65535 may be defined by base station manufacturers. The following service option numbers have been assigned or are reserved for the noted service.

Table A-1. Service Option Numbers

Service Option Number	Service
1	Variable Data Rate Two-Way Voice
2	Reserved for Test Option 1
3	Reserved for Data Option 1
4	Reserved for FAX Option 1
5	Reserved for Alternate Vocoder 1

A.1 Service Option 1: Variable Data Rate Two-Way Voice

A.1.1 General Description

Service Option 1 provides two-way voice communications between the base station and the mobile station using the dynamically variable data rate vocoder algorithm described in this appendix. The transmitting vocoder takes voice samples and generates, for every Traffic Channel frame, an encoded speech packet for transmission to the receiving vocoder. For every Traffic Channel frame, the receiving vocoder decodes the received speech packet into voice samples.

The two vocoders communicate at one of four rates corresponding to the 9600 bps, 4800 bps, 2400 bps, and 1200 bps frame rates.

A.1.2 Service Option Number

The variable data rate two-way voice service option using the dynamically variable data rate vocoder algorithm described herein shall use service option number 1 and is called Service Option 1.

A.1.3 Multiplex Option

A.1.3.1 Required Multiplex Option

Service Option 1 shall only be used with Multiplex Option 1. Service Option 1 shall only be connected as primary traffic.

A.1.3.2 Interface to Multiplex Option 1

A.1.3.2.1 Transmitted Traffic Channel Frames

For every transmitted Traffic Channel frame,[1] the vocoder shall generate and supply one packet to the multiplex sublayer. The packet contains the service option information bits which are transmitted as primary traffic. The packet shall be one of five types as shown in Table A.1.3.2.1-1. The number of bits supplied to the multiplex sublayer for each type of packet shall also be as shown in Table A.1.3.2.1-1. Four of the five packet types are called Rate 1, Rate 1/2, Rate 1/4, and Rate 1/8. Upon command, the vocoder shall generate a fifth type of packet, called a blank packet, that supplies no bits to the multiplex sublayer. A blank packet is used for blank-and-burst transmission of signaling traffic (see 6.1.3.3.11). Upon command, the vocoder shall generate other than a Rate 1 packet. Under any other condition, it is free to determine whether to supply either a Rate 1, a Rate 1/2, a Rate 1/4, or a Rate 1/8 packet to the multiplex sublayer.

Table A.1.3.2.1-1. Packet Types Supplied by Service Option 1 to the Multiplex Sublayer

Packet Type	Bits per Packet
Rate 1	171
Rate 1/2	80
Rate 1/4	40
Rate 1/8	16
Blank	0

[1]The CAI uses the term frame to represent a 20 ms grouping of data on the Traffic Channel. Common vocoder terminology also uses the term frame to represent a quantum of processing. For Service Option 1, the vocoder frame corresponds to speech sampled over 20 ms. The 20 ms speech samples are processed into a packet. This packet is transmitted in a Traffic Channel frame. In most cases, the two types of frames will not be confused. In cases where they may be confused, a frame will be explicitly called either a vocoder frame or a Traffic Channel frame.

A.1.3.2.2 Received Traffic Channel Frames

The multiplex sublayer in the mobile station categorizes every received Traffic Channel frame (see 6.2.2.2), and supplies the packet type and accompanying bits, if any, to the vocoder as shown in Table A.1.3.2.2-1. The vocoder processes the bits of the packet as described in A.1.4. The first five received packet types shown in Table A.1.3.2.2-1 correspond to the transmitted packet types shown in Table A.1.3.2.1-1. The blank packet occurs when the receiving station determines that a blank-and-burst frame for signaling traffic or secondary traffic was transmitted. The Rate 1 packet with probable bit errors category occurs when the receiving station determines that the frame was transmitted at 9600 bps and the frame is likely to have one or more bit errors. The last category occurs when the quality of the received frame is insufficient to decide upon the rate.

Table A.1.3.2.2-1. Packet Types Supplied by the Multiplex Sublayer to Service Option 1

Packet Type	Bits per Packet
Rate 1	171
Rate 1/2	80
Rate 1/4	40
Rate 1/8	16
Blank	0
Rate 1 with probable bit errors	171
Insufficient frame quality (erasure)	0

A.1.3.3 Connection and Initialization

For a mobile station originated call, the mobile station connects and initializes Service Option 1 and begins transferring packets between the multiplex sublayer and Service Option 1 when it enters the *Conversation Substate* of the *Mobile Station Control on the Traffic Channel State* (see 6.6.4.4). Service Option 1 shall be initialized as described in A.1.4.9.

For a mobile station terminated call, the mobile station connects and initializes the receiving side of Service Option 1 when it enters the *Waiting for Order Substate* of the *Mobile Station Control on the Traffic Channel State*. The mobile station then begins transferring packets from the multiplex sublayer to Service Option 1 (see 6.6.4.3.1). The mobile station connects and initializes the transmitting side of Service Option 1 when it enters the *Conversation Substate* of the *Mobile Station Control on the Traffic Channel State*. At this time, it begins transferring packets from Service Option 1 to the multiplex sublayer. The initializations are described in A.1.4.9.

A.1.3.4 Service Option Control Orders

The base station may send the *Service Option Control Order* to the mobile station on the Forward Traffic Channel (see 7.7.4). The mobile station does not send the *Service Option Control Order*. The mobile station shall allow at least one *Service Option Control Order* with a specified ACTION_TIME for this service option.[2]

If the ORDQ field in a *Service Option Control Order* referring to Service Option 1 equals '00000001', then the mobile station shall initialize both the transmitting and receiving of the vocoder as described in A.1.4.9. The initializations shall be performed within 40 ms (USE_TIME equals '0') or within 40 ms of the time specified by the ACTION_TIME field (USE_TIME equals '1'). If the ORDQ field in a *Service Option Control Order* referring to Service Option 1 equals '00000010', then the mobile station performs the following two actions:[3] When the mobile station initializes the transmitting side of the vocoder, it should disable the audio output of the vocoder for 1 second. The mobile station shall process a blank packet as an insufficient frame quality (erasure) packet. Any other *Service Option Control Order* referring to Service Option 1 and having an ORDQ field other than '00000001' or '00000010' shall be rejected using the *Mobile Station Reject Order* with an ORDQ field equal to '00000100' (see Table 6.7.3-1).

A.1.3.5 Service Option Negotiation

If the mobile station receives a *Service Option Request Order* referring to Service Option 1, it shall respond within 200 ms either accepting the service option, rejecting the service option, or suggesting an alternative service option (see 6.6.4.1.2.2.1).

If the base station receives a *Service Option Request Order* referring to Service Option 1, it shall respond within 5 seconds either accepting the service option, rejecting the service option, or suggesting an alternative service option (see 7.6.4.1.2.2.1).

A.1.4 Variable Rate Speech Coding Algorithm

A.1.4.1 Introduction

The vocoder uses a code excited linear predictive (CELP) coding algorithm. This technique uses a random codebook to vector quantize the residual signal using an analysis-by-synthesis method. The vocoder produces a variable output data rate based on 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 below in Figure A.1.4.1-1. First a vector, specified by $\hat{I}$, is taken from a codebook of random Gaussian vectors (for Rate 1/8, a random vector is generated). This vector is multiplied by a gain term $\hat{G}$, and then is filtered

---

[2]The mobile station may have other pending messages and orders.

[3]This capability is to support mobile station to mobile station calls which do not use tandem vocoding.

by the long-term pitch filter whose characteristics are governed by the pitch parameters $\hat{L}$ and $\hat{b}$. This output is filtered by the formant synthesis filter, also called the linear predictive coding filter, to reproduce a speech signal. The speech signal is filtered by the adaptive postfilter.

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

The vocoder 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 generated codebook vector as shown in Figure A.1.4.1-1.

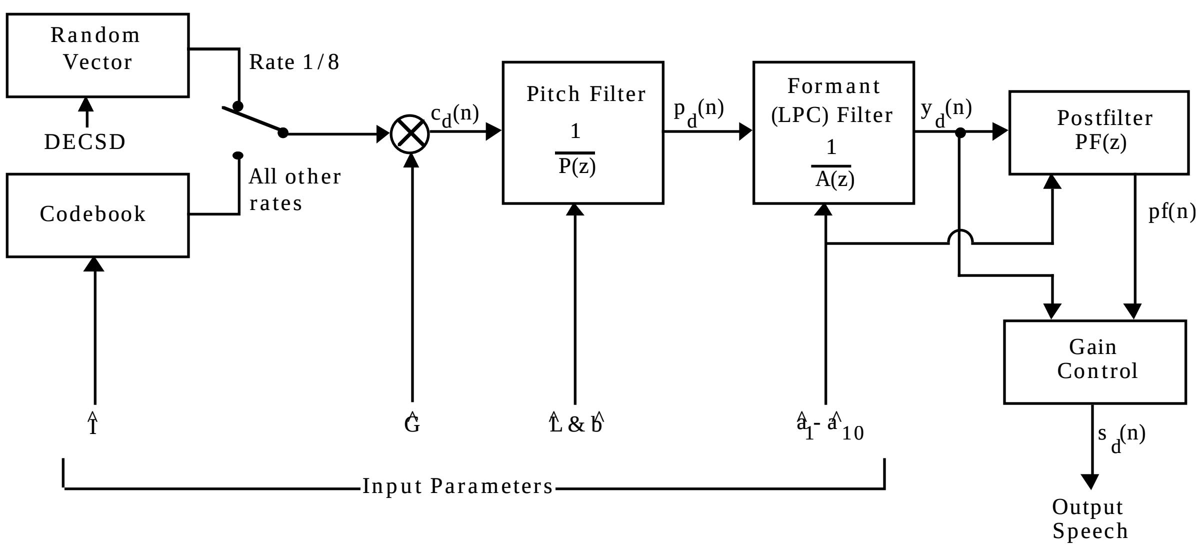

Figure A.1.4.1-1. Speech Synthesis Structure in the Receiving Vocoder

The input speech is sampled at 8 kHz. This speech is broken down into 20 ms vocoder frames consisting of 160 samples. The linear predictive coding (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 frame, the pitch parameters are updated a varying number of times, where the number of pitch parameter updates is also a function of the selected data rate. Similarly, the codebook parameters are updated a varying number of times, again where the number of updates is a function of the selected data rate. Table A.1.4.1-1 describes the various parameters used for each rate.

Table A.1.4.1-1. 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	40	20	10	10
Pitch updates (subframes) per frame	4	2	1	0
Samples per pitch subframe, L_P	40 (5 ms)	80 (10 ms)	160 (20 ms)	–
Bits per pitch update	10	10	10	–
Codebook updates (subframes) per frame	8	4	2	1
Samples per codebook subframe, L_C	20 (2.5 ms)	40 (5 ms)	80 (10 ms)	160 (20 ms)
Bits per codebook update	10	10	10	6*
*Note: Rate 1/8 uses six bits for pseudorandom excitation, rather than using the codebook.				

The hierarchy of frames and subframes for each rate is shown in Figures A.1.4.1-2 through A.1.4.1-5. In these figures, each large block represents 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 ten bits to encode the new pitch parameters. This is done a varying number of times for the other rates as shown. Note that a pitch update is not done at Rate 1/8, as this rate is used to encode frames when little or no speech is present and pitch redundancies do not exist. 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, the codebook parameters are updated eight times, once for each eighth of the speech frame, each time using ten bits to encode the parameters. The number of updates decreases as the rate decreases.

LPC Frame	40								Total = 160 bits
Pitch Subframe	10		10		10		10		(plus 11 parity
Codebook Subframe	10	10	10	10	10	10	10	10	check bits)

Figure A.1.4.1-2. Rate 1 Bit Allocation for a Vocoder Frame

LPC Frame	20				
Pitch Subframe	10		10		Total = 80 bits
Codebook Subframe	10	10	10	10	

Figure A.1.4.1-3. Rate 1/2 Bit Allocation for a Vocoder Frame

LPC Frame	10		
Pitch Subframe	10		Total = 40 bits
Codebook Subframe	10	10	

Figure A.1.4.1-4. Rate 1/4 Bit Allocation for a Vocoder Frame

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

Figure A.1.4.1-5. Rate 1/8 Bit Allocation for a Vocoder Frame

Table A.1.4.1-2 lists all the parameter codes transmitted for each rate packet. The following list describes each parameter:

LSPi	Line Spectral Pair frequency i.
PLAGi	Pitch Lag for the ith pitch subframe.
PGAINi	Pitch Gain for the ith pitch subframe.
CBINDEXi	Codebook Index for the ith codebook subframe.
CBGAINi	Codebook Gain for the ith codebook subframe.
CBSEED	Random Seed for Rate 1/8 packets.
PCB	Parity Check Bits used to detect and correct errors in a Rate 1 packet.

This appendix 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 LSP1 = ‘1011’ in binary for a maximum rate frame, LSP1[0] = ‘1’, LSP1[1] = ‘1’, LSP1[2] = ‘0’, and LSP1[3] = ‘1’.

Table A.1.4.1-2. Transmission Codes and Bit Allocations

	Rate					Rate			
Code	**1**	**1/2**	**1/4**	**1/8**	**Code**	**1**	**1/2**	**1/4**	**1/8**
LSP1	4	2	1	1	CBINDEX1	7	7	7	-
LSP2	4	2	1	1	CBINDEX2	7	7	7	-
LSP3	4	2	1	1	CBINDEX3	7	7	-	-
LSP4	4	2	1	1	CBINDEX4	7	7	-	-
LSP5	4	2	1	1	CBINDEX5	7	-	-	-
LSP6	4	2	1	1	CBINDEX6	7	-	-	-
LSP7	4	2	1	1	CBINDEX7	7	-	-	-
LSP8	4	2	1	1	CBINDEX8	7	-	-	-
LSP9	4	2	1	1	CBGAIN1	3	3	3	2
LSP10	4	2	1	1	CBGAIN2	3	3	3	-
PLAG1	7	7	7	-	CBGAIN3	3	3	-	-
PLAG2	7	7	-	-	CBGAIN4	3	3	-	-
PLAG3	7	-	-	-	CBGAIN5	3	-	-	-
PLAG4	7	-	-	-	CBGAIN6	3	-	-	-
PGAIN1	3	3	3	-	CBGAIN7	3	-	-	-
PGAIN2	3	3	-	-	CBGAIN8	3	-	-	-
PGAIN3	3	-	-	-	CBSEED	-	-	-	4
PGAIN4	3	-	-	-	PCB	11	-	-	-

A.1.4.2 Input Audio Interface

A.1.4.2.1 Input Audio Interface in the Mobile Station

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

A.1.4.2.1.1 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 bits of dynamic range.

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

A.1.4.2.1.2 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.[4]

A.1.4.2.1.3 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 vocoder 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.

A.1.4.2.1.3.1 Transmit Level Adjustment

Pending the generation of a complete speech transmission plan for dual-mode cellular systems, the following requirements shall be met to ensure compatibility with the transmission plan for fixed digital speech networks.

The mobile station shall have a transmit objective loudness rating (TOLR) equal to -46 dB, when transmitting to a reference base station (see A.1.4.10.2.1.3). The loudness ratings are described in IEEE Standard 661-1979.[5] Measurement techniques are described in "Recommended Minimum Performance Standards for 800 MHz Wideband Spread Spectrum Dual-Mode Mobile Stations."

A.1.4.2.1.3.2 Band Pass Filtering

Input anti-aliasing filtering shall conform to CCITT Recommendation G.714.[6] Additional anti-aliasing filtering may be provided by the manufacturer.

A.1.4.2.1.3.3 Echo Return Loss

Provision shall be made to ensure adequate isolation between receive and transmit audio paths in all modes of operation. The receive audio at full volume shall not couple into the transmit audio path so that the vocoder generates other than Rate 1/8 packets (see A.1.4.4) when no external audio is present. Refer to the requirements stated in "Recommended Minimum Performance Standards for 800 MHz Wideband Spread Spectrum Dual-Mode Mobile Stations."

---

[4] See CCITT Recommendation "Pulse Code Modulation (PCM) of Voice Frequencies," Vol III, Recommendation G.711, Geneva 1972.

[5] See "IEEE Standard Method for Determining Objective Loudness Ratings of Telephone Connections," ANSI/IEEE Standard 661-1979.

[6] See CCITT Recommendation "Separate Performance Characteristics for the Encoding and Decoding Sides of PCM Channels Applicable to 4-Wire Voice-Frequency Interfaces," Blue Book, Vol III, Recommendation G.714, Melbourne, 1988.

A.1.4.2.2 Input Audio Interface in the Base Station

A.1.4.2.2.1 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 bits of dynamic range. The sampling rate is 8000 samples per second. The sampling and conversion process shall be as in A.1.4.2.1.

A.1.4.2.2.2 Transmit Level Adjust

Pending the generation of a complete speech transmission plan for dual-mode cellular systems, the following requirements shall be met to ensure compatibility with the transmission plan for fixed digital speech networks.

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 maximum amplitude at the output of the quantizer. Measurement techniques are described in "Recommended Minimum Performance Standards for 800 MHz Base Stations Supporting Wideband Spread Spectrum Dual-Mode Mobile Stations."

A.1.4.2.2.3 Echo Canceling

The base station shall provide a method to cancel echoes returned by the PSTN interface.[7] 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.

A.1.4.3 Determining the Formant Prediction Parameters

A.1.4.3.1 Form of the Formant Synthesis Filter

The formant synthesis filter is equivalent to the traditional LPC formant synthesis filter. The transfer function for the formant prediction error filter, which removes the short term redundancies in the speech, is[8]

---

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

[8]Because of the large number of mathematical equations, this appendix uses the implied multiplication operator rather than the explicit operator "×" as is used in most of this document.

This appendix uses the two-sided z-transform as defined by (see Oppenheim, A. V. and Schafer, R. W., *Digital Signal Processing*, (New Jersey: Prentice-Hall Inc., 1975), pp. 45 - 86):

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

$$A(z) = 1 - \sum_{i=1}^{P} a_i z^{-i} \quad . \qquad \text{(A.1.4.3.1-1)}$$

The filter is a tenth-order filter (i.e. P equals 10). The formant synthesis filter, which reinserts the redundancies at the receiving end, is given by the inverse of A(z):

$$\frac{1}{A(z)} = \frac{1}{1 - \sum_{i=1}^{P} a_i z^{-i}} \quad . \qquad \text{(A.1.4.3.1-2)}$$

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

A.1.4.3.2 Encoding

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

1. Remove the DC from the input samples.
2. Window the input samples using a Hamming window.
3. Compute the autocorrelation function for 11 lags.
4. Determine the LPC coefficients from the autocorrelation values.
5. Bandwidth expand the LPC coefficients.
6. Transform the scaled coefficients to LSP frequencies.
7. Convert the LSP frequencies into LSP codes
   (these codes are placed into the packet for transmission).

A.1.4.3.2.1 Removing the DC Component

A DC block is inserted to prevent a DC offset from artificially increasing R(0) (see A.1.4.3.2.3) and thus disrupting the rate decision algorithm (see A.1.4.4).[9]

A.1.4.3.2.2 Windowing the Samples

The coefficients are computed from a Hamming window of speech centered at the center of the fourth Rate 1 pitch subframe. The window is 160 samples long (i.e., L_A equals 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

---

[9]One of several such methods would be to take the average of the 160 samples in the current window of speech, low pass filter this average to prevent large discontinuities at the frame boundaries, and subtract this low passed filtered average from the 160 samples in the current window.

$$s_W(n) = s(n + 60)\, W_H(n) \qquad \text{for } 0 \le n \le L_A - 1 \qquad \text{(A.1.4.3.2.2-1)}$$

where the Hamming window is defined as

$$W_H(n) = \begin{cases} 0.54-0.46\cos\left(\dfrac{2\pi n}{L_A-1}\right) & \text{for } 0 \le n \le L_A-1 \\ 0 & \text{for all other } n. \end{cases} \qquad \text{(A.1.4.3.2.2-2)}$$

Note the offset of 60 samples, which results in the window of speech being centered between the 139th and 140th sample of the current 160 sample frame of speech.

A.1.4.3.2.3 Computing the Autocorrelation Function

Following the windowing operation, the kth autocorrelation coefficient is computed as

$$R(k) = \sum_{m=0}^{L_A-1-k} s_W(m)\, s_W(m + k)\ . \qquad \text{(A.1.4.3.2.3-1)}$$

Only the first 11 autocorrelation coefficients R(0) through R(10) need be computed from the windowed speech signal within the analysis window.

A.1.4.3.2.4 Determining the LPC Coefficients from the Autocorrelation Function

Next, the LPC coefficients are obtained from the autocorrelation function. A method is Durbin's recursion, described on the following page.[10]

---

[10]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.

{
$E^{(0)} = R(0)$
$i = 1$
while $(i \le P)$
{
$$k_i = \left\{ R(i) - \sum_{j=1}^{i-1} \alpha_j^{(i-1)} R(i-j) \right\} / E^{(i-1)}$$
$\alpha_i^{(i)} = k_i$
$j = 1$
while $(j \le i-1)$
{
$\alpha_j^{(i)} = \alpha_j^{(i-1)} - k_i \alpha_{i-j}^{(i-1)}$
$j = j + 1$
}
$E^{(i)} = (1 - k_i^2) E^{(i-1)}$
$i = i + 1$
}
}

The LPC coefficients before bandwidth expansion are $\alpha_j^{(P)}$, where $1 \le j \le P$.

A.1.4.3.2.5 Expanding the Bandwidth

Next, the LPC coefficients have 15 Hz of bandwidth expansion applied before they are transformed into LSP frequencies. This is done by scaling the poles of the formant synthesis filter radially inwards. Each LPC coefficient, $\alpha_i^{(P)}$, is scaled by $ß^i$ (ß to the ith power) as follows:

$$a_i = ß^i \alpha_i^{(P)} \qquad 1 \le i \le P \qquad \text{(A.1.4.3.2.5-1)}$$

where ß is 0.9883.

A.1.4.3.2.6 Transforming the LPC Coefficients to Line Spectrum Pairs (LSPs)

Next, the LPC coefficients are transformed into line spectrum pair frequencies. The basic computation of the LSP frequencies follows.

As before, A(z) is given by

$$A(z) = 1 - a_1 z^{-1} - \ldots - a_{10} z^{-10} \qquad \text{(A.1.4.3.2.6-1)}$$

where a_i $(1 \le i \le 10)$ are the LPC coefficients as described above.

Define $P_A(z)$ and $Q_A(z)$ as follows:

$$P_A(z) = A(z) + z^{-11}A(z^{-1}) = 1 + p_1z^{-1} + \ldots + p_5z^{-5} + p_5z^{-6} + \ldots + p_1 z^{-10} + z^{-11}. \quad \text{(A.1.4.3.2.6-2)}$$

$$Q_A(z) = A(z) - z^{-11}A(z^{-1}) = 1 + q_1z^{-1} + \ldots + q_5z^{-5} - q_5z^{-6} - \ldots - q_1z^{-10} - z^{-11}. \quad \text{(A.1.4.3.2.6-3)}$$

where $p_i = - a_i - a_{11-i} \qquad 1 \le i \le 5$ (A.1.4.3.2.6-4)

$$q_i = - a_i + a_{11-i} \qquad 1 \le i \le 5. \quad \text{(A.1.4.3.2.6-5)}$$

The LSP frequencies are the ten roots which exist between w = 0 and w = 0.5 in the following two equations:

$$P'(w) = \cos 5(2\pi w) + p'_1 \cos 4(2\pi w) + \ldots + p'_4 \cos (2\pi w) + p'_5/2 \quad \text{(A.1.4.3.2.6-6)}$$

$$Q'(w) = \cos 5(2\pi w) + q'_1 \cos 4(2\pi w) + \ldots + q'_4 \cos (2\pi w) + q'_5/2 \ , \quad \text{(A.1.4.3.2.6-7)}$$

where the p' and q' values are computed recursively as follows from the p and q values defined above.

$$p'_0 = q'_0 = 1 \quad \text{(A.1.4.3.2.6-8)}$$

$$p'_i = p_i - p'_{i-1} \qquad 1 \le i \le 5. \quad \text{(A.1.4.3.2.6-9)}$$

$$q'_i = q_i + q'_{i-1} \qquad 1 \le i \le 5. \quad \text{(A.1.4.3.2.6-10)}$$

A property of the LSP frequencies is that if the LPC filter is stable, the roots of the two functions alternate; the smallest root, w_1, is the lowest root of P'(w), the next smallest root, w_2, is the lowest root of Q'(w), etc. Thus, w_1, w_3, w_5, w_7, and w_9, are the roots of P'(w), and w_2, w_4, w_6, w_8, and w_{10} are the roots of Q'(w).

A.1.4.3.2.7 Converting the LSP Frequencies to Transmission Codes

Once the LSP frequencies have been computed and the data rate has been selected (see A.1.4.4), each LSP frequency is converted for transmission. The converter is shown in Figure A.1.4.3.2.7-1.

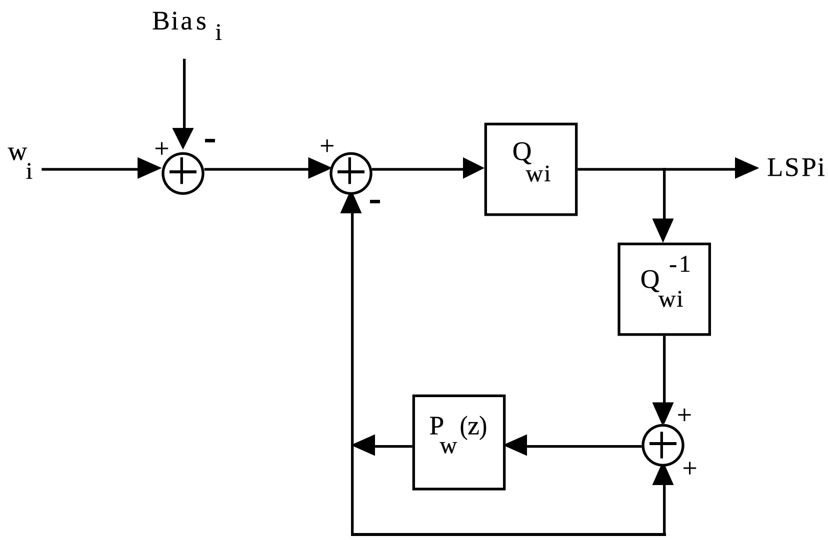

Figure A.1.4.3.2.7-1. Converting the LSP Frequencies to Transmission Codes

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 as follows:

$$\mathrm{Bias}_i = \frac{0.5i}{P+1} = (0.04545...)\, i \qquad 1 \le i \le 10. \tag{A.1.4.3.2.7-1}$$

where P is the order of the predictor and is equal to 10.

The predictor P_w is

$$P_w(z) = 0.90625\, z^{-1}\,. \tag{A.1.4.3.2.7-2}$$

The predictor is updated once per frame unless a blank packet has been requested. There is one predictor for each LSP frequency.

The quantizer, Q_{wi}, for the ith LSP frequency is a linear quantizer which varies in dynamic range and step size with rate. Each LSP frequency is quantized as follows:

$$Q_{wi}(x) = \max\,[0, \min\,(2^N - 1, Q_{ti}(x))] \tag{A.1.4.3.2.7-3}$$

where

$$Q_{ti}(x) = \mathrm{round}\left(\frac{2^N - 1}{2}\, \frac{x + Q_{wi}^{max}}{Q_{wi}^{max}} \right), \tag{A.1.4.3.2.7-4}$$

N is the number of bits of quantization, Q_{wi}^{max} is the maximum quantization level given for the ith coefficient, and round (x) is the function rounding to the closest integer. The number of LSP quantization bits, N, is given in Table A.1.4.3.2.7-1. The maximum quantization level, Q_{wi}^{max}, is given in Table A.1.4.3.2.7-2. Note that Equation A.1.4.3.2.7-3 is a limiting function to maintain $Q_{wi}(x)$ between 0 and 2^N - 1.

Table A.1.4.3.2.7-1. Number of LSP Quantization Bits

LSP Frequency	Rate 1	Rate 1/2	Rate 1/4	Rate 1/8
w_1	4	2	1	1
w_2	4	2	1	1
w_3	4	2	1	1
w_4	4	2	1	1
w_5	4	2	1	1
w_6	4	2	1	1
w_7	4	2	1	1
w_8	4	2	1	1
w_9	4	2	1	1
w_{10}	4	2	1	1
Total	**40**	**20**	**10**	**10**

Table A.1.4.3.2.7-2. Maximum LSP Quantization Level

LSP Frequency	Rate 1	Rate 1/2	Rate 1/4	Rate 1/8
w_1	0.025	0.015	0.01	0.01
w_2	0.04	0.015	0.01	0.01
w_3	0.07	0.03	0.01	0.01
w_4	0.07	0.03	0.01	0.01
w_5	0.06	0.03	0.01	0.01
w_6	0.06	0.02	0.01	0.01
w_7	0.05	0.02	0.01	0.01
w_8	0.05	0.02	0.01	0.01
w_9	0.04	0.02	0.01	0.01
w_{10}	0.04	0.02	0.01	0.01

A.1.4.3.3 Decoding

The decoding process consists of the following steps:

1. Convert the LSP transmission codes to LSP frequencies.
2. Check the stability of the LSP frequencies.
3. Low-pass filter the LSP frequencies.
4. Interpolate the LSP frequencies.
5. Convert the interpolated LSP frequencies to LPC coefficients.

The steps taken by the receiving decoder (see A.1.4.11.2) are similar to the transmitting vocoder except for the possibility of receiving a packet type equal to insufficient frame quality (see A.1.3.2.2).

A.1.4.3.3.1 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 compute the actual LSP frequencies, $\tilde{w}_i$ (see Figure A.1.4.3.3.1-1).

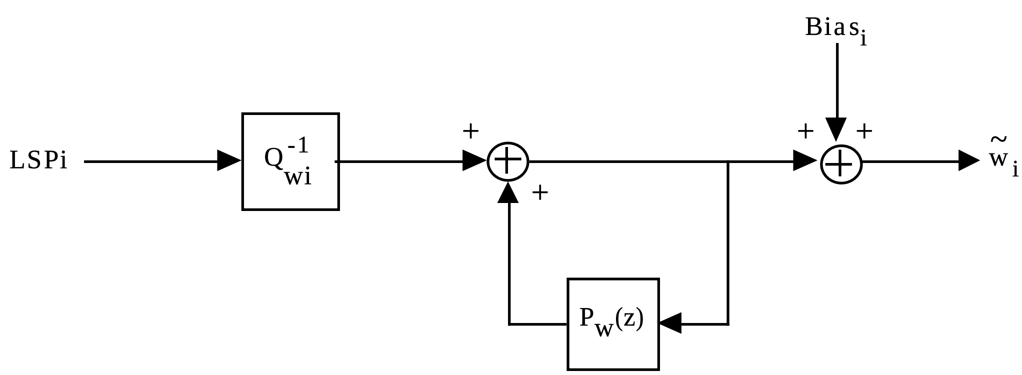

Figure A.1.4.3.3.1-1. Converting the LSP Transmission Codes to LSP Frequencies

The predictor $P_w(z)$ is the same as in Equation A.1.4.3.2.7-2. The predictor is updated for every packet except for the blank packet. The bias is given in Equation A.1.4.3.2.7-1. The quantizer is the inverse of that given by Equation A.1.4.3.2.7-4.

A.1.4.3.3.2 Checking the Stability of the LSP Frequencies

Before converting the LSP frequencies back to LPC coefficients, a check is done to ensure that the resulting filter has not been made unstable due to quantization noise or channel errors injecting noise into one or many frequencies. Stability is guaranteed if the LSP frequencies remain ordered. In addition, the frequencies are forced to be at least 80 Hz apart to prevent unusually large peaks in the formant filter response. This ordering and minimum spacing are enforced using the following algorithm:

$$
\begin{array}{l}
\{ \\
\quad \tilde{w}_0 = 0.0 \\
\quad i = 0 \\
\quad \text{while } (i < 10) \\
\qquad \{ \\
\qquad\quad \text{if } ((\tilde{w}_{i+1} - \tilde{w}_i) < \Delta\tilde{w}_{min}) \\
\qquad\qquad \tilde{w}_{i+1} = \tilde{w}_i + \Delta\tilde{w}_{min} \\
\qquad\quad i = i + 1 \\
\qquad \} \\
\quad \tilde{w}_{11} = 0.5 \\
\quad \text{while } (i > 0) \\
\qquad \{ \\
\qquad\quad \text{if } ((\tilde{w}_{i+1} - \tilde{w}_i) < \Delta\tilde{w}_{min}) \\
\qquad\qquad \tilde{w}_i = \tilde{w}_{i+1} - \Delta\tilde{w}_{min} \\
\qquad\quad i = i - 1 \\
\qquad \} \\
\}
\end{array}
$$

A $\Delta\tilde{w}_{min}$ of 0.01 is used. This results in 80 Hz separation in the LSP domain.

A.1.4.3.3.3 Low-Pass Filtering the LSP Frequencies

Next, the LSP frequencies are low-pass filtered as follows to remove some of the quantization noise effects at lower rates:

$$\hat{w}_i(\text{current frame}) = SM\ \hat{w}_i\ (\text{previous frame}) + (1-SM)\ \tilde{w}_i(\text{current frame})\ . \qquad \text{(A.1.4.3.3.3-1)}$$

The value of SM depends upon the packet rate. For both the encoder and decoder, a counter is used to track the number of consecutive packets that are either Rate 1/4 or Rate 1/8. If the current packet is either Rate 1/4 or Rate 1/8, the counter is incremented. If the current packet is either Rate 1 or Rate 1/2, the counter is set to zero. Otherwise the counter is unchanged. The value of SM that is used in Equation A.1.4.3.3.3-1 is given in Equation A.1.4.3.3.3-2. A received packet categorized as Rate 1 with probable bit errors is treated as a Rate 1 packet if the packet is detected as having one or fewer errors; otherwise the packet is treated as an erasure (see A.1.4.8.6.3).

$$SM = \begin{cases} 0 & \text{if packet is Rate 1} \\ 0.125 & \text{if packet is Rate 1 / 2} \\ 0.125 & \text{if packet is Rate 1 / 4 or 1 / 8 and counter } < 10 \\ 0.9 & \text{if packet is Rate 1 / 4 or 1 / 8 and counter } \geq 10 \\ 0.875 & \text{if an insufficient frame quality packet (erasure)} \end{cases} \qquad \text{(A.1.4.3.3.3-2)}$$

A.1.4.3.3.4 Interpolating the LSP Frequencies

Next, the LSP frequencies are interpolated for each subframe of the pitch and codebook searches in the selected rate.

In calculating the original LPC coefficients, a speech window centered between the 139th and 140th sample of the frame was used. In performing the pitch and codebook searches for the smaller subframes, LPC coefficients which are accurate at the center of the particular pitch subframe should be used (except at Rate 1/8, where it is the center of the single codebook subframe). 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 A.1.4.3.3.4-1. In all cases $\hat{\omega}_i$(previous) is the ith filtered LSP frequency from the previous frame and $\hat{\omega}_i$(current) is the ith filtered LSP frequency from the current frame.

Table A.1.4.3.3.4-1. LSP Subframe Interpolation for All Rates

Rate 1	**For pitch subframe**	**For codebook subframes**
$\hat{\omega}'_i = 0.75\ \hat{\omega}_i(\text{previous}) + 0.25\ \hat{\omega}_i(\text{current})$	1	1 and 2
$\hat{\omega}'_i = 0.5\ \hat{\omega}_i(\text{previous}) + 0.5\hat{\omega}_i(\text{current})$	2	3 and 4
$\hat{\omega}'_i = 0.25\ \hat{\omega}_i(\text{previous}) + 0.75\ \hat{\omega}_i(\text{current})$	3	5 and 6
$\hat{\omega}'_i = \hat{\omega}_i(\text{current})$	4	7 and 8

Rate 1/2	**For pitch subframe**	**For codebook subframe**
$\hat{\omega}'_i = 0.625\ \hat{\omega}_i(\text{previous}) + 0.375\ \hat{\omega}_i(\text{current})$	1	1 and 2
$\hat{\omega}'_i = 0.125\ \hat{\omega}_i(\text{previous}) + 0.875\ \hat{\omega}_i(\text{current})$	2	3 and 4

Rate 1/4	**For pitch subframe**	**For codebook subframe**
$\hat{\omega}'_i = 0.375\ \hat{\omega}_i(\text{previous}) + 0.625\ \hat{\omega}_i(\text{current})$	1	1 and 2

Rate 1/8	**For pitch subframe**	**For codebook subframe**
$\hat{\omega}'_i = 0.375\ \hat{\omega}_i(\text{previous}) + 0.625\ \hat{\omega}_i(\text{current})$	–	1

A.1.4.3.3.5 Converting the Interpolated LSP Frequencies to LPC Coefficients

Next, the interpolated LSP frequencies are converted back into LPC coefficients for use in the pitch and codebook searches. In addition, the converted LSP frequencies are used by the receiving decoder for speech generation as described in A.1.4.11.2. The conversion method is as follows:

First, $\hat{P}_A(z)$ and $\hat{Q}_A(z)$ are computed from the LSP frequencies using Equations A.1.4.3.3.5-1 and A.1.4.3.3.5-2:

$$\hat{P}_A(z) = (1 + z^{-1}) \prod_{j=1}^{5} (1-2z^{-1}\cos(2\pi \hat{w}'_{(2j-1)})+z^{-2}) \qquad \text{(A.1.4.3.3.5-1)}$$

and

$$\hat{Q}_A(z) = (1 - z^{-1}) \prod_{j=1}^{5} (1-2z^{-1}\cos(2\pi \hat{w}'_{(2j)})+z^{-2}) . \qquad \text{(A.1.4.3.3.5-2)}$$

Then the LPC coefficients are computed from the coefficients of $\hat{P}_A(z)$ and $\hat{Q}_A(z)$ as follows:

$$A(z) = \frac{\hat{P}_A(z)+\hat{Q}_A(z)}{2}$$

$$= 1 + \frac{(\hat{p}_1+\hat{q}_1)}{2} z^{-1} + \ldots + \frac{(\hat{p}_5+\hat{q}_5)}{2} z^{-5} + \frac{(\hat{p}_5-\hat{q}_5)}{2} z^{-6} + \ldots + \frac{(\hat{p}_1-\hat{q}_1)}{2} z^{-10}$$

$$= 1 - \hat{a}_1 z^{-1} - \ldots - \hat{a}_{10} z^{-10} \qquad \text{(A.1.4.3.3.5-3)}$$

so

$$\hat{a}_i = \begin{cases} -\dfrac{\hat{p}_i+\hat{q}_i}{2} & 1 \le i \le 5 \\ -\dfrac{\hat{p}_{11-i}-\hat{q}_{11-i}}{2} & 6 \le i \le 10 \end{cases} \qquad \text{(A.1.4.3.3.5-4)}$$

The LPC coefficients for the particular subframe are the $\hat{a}_i$ defined in Equation A.1.4.3.3.5-4.

A.1.4.4 Determining the Data Rate

A.1.4.4.1 Threshold Comparing

Unless specifically requested to generate a blank packet or not to generate a Rate 1 packet, the vocoder is free to generate a packet at any of the four rates. Three thresholds are maintained as described in A.1.4.4.2. When the vocoder is free to choose the rate, it is based on the energy in the frame and the threshold described in A.1.4.4.2. The energy in the frame is estimated by $R_i(0)$, the first autocorrelation coefficient for the ith frame which is defined in Equation A.1.4.3.2.3-1. The threshold is based upon an estimate of the background noise level B_i, computed for the ith frame.

$R_i(0)$ is compared with the three thresholds: $T_1(B_i)$, $T_2(B_i)$, and $T_3(B_i)$. If $R_i(0)$ is greater than all three thresholds, Rate 1 is selected. If $R_i(0)$ is greater than only two thresholds, Rate 1/2 is selected. If $R_i(0)$ is greater than only one threshold, Rate 1/4 is selected. If $R_i(0)$ is below all three thresholds, Rate 1/8 is selected.

Several constraints are placed upon the selected data rate. First, the data rate is only permitted to decrease by one rate per frame. If the previous frame was encoded at Rate 1 and the initial rate selection for the current frame is Rate 1/4 or Rate 1/8, then Rate 1/2 is chosen. Similarly, if the previous frame was encoded at Rate 1/2 and the initial selection for the current frame is Rate 1/8, then Rate 1/4 is chosen.

Second, if the vocoder has been commanded not to generate a Rate 1 packet, it generates a Rate 1/2 packet if the rate determined by the threshold tests is Rate 1. Third, if the vocoder has been told to generate a blank packet, it generates a blank packet regardless of the rate determined by the threshold tests.

A.1.4.4.2 Updating Thresholds

The three thresholds are updated every frame before the rate is determined. First, an estimate of the background noise level B_i is computed for the current or ith frame as follows:

$$B_i = \min\,[R_{i-1}(0),\ 160000,\ \max\,(1.00547B_{i-1},\ B_{i-1} + 1)] \qquad \text{(A.1.4.4.2-1)}$$

where min (x,y,z) is the minimum of x, y, and z, and max (x,y) is the maximum of x and y.

At initialization, the background noise estimate for the first frame, B_1, is set to 160000. If the audio input to the encoder is disabled, the background noise estimate is reinitialized whenever the audio is re-enabled.[11]

---

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

Then the three thresholds are computed as a function of B_i as follows:

$$T_1(B_i) = -(5.544613 \times 10^{-6})\, B_i^2 + 4.047152\, B_i + 362$$

$$T_2(B_i) = -(1.529733 \times 10^{-5})\, B_i^2 + 8.750045\, B_i + 1136$$

$$T_3(B_i) = -(3.957050 \times 10^{-5})\, B_i^2 + 18.89962\, B_i + 3347\ . \qquad \text{(A.1.4.4.2-2)}$$

A.1.4.5 Determining the Pitch Prediction Parameters

A.1.4.5.1 Encoding

All vocoder frames, except for frames being encoded into Rate 1/8 packets, are subdivided into equal length pitch subframes as shown in Figures A.1.4.1-2 through A.1.4.1-5 and Table A.1.4.1-1. There are four pitch subframes for a Rate 1 packet, two pitch subframes for a Rate 1/2 packet, and one pitch subframe for a Rate 1/4 packet. There are no pitch subframes for a Rate 1/8 packet. The pitch synthesis filter can be expressed as:

$$\frac{1}{P(z)} = \frac{1}{1-bz^{-L}}\ . \qquad \text{(A.1.4.5.1-1)}$$

The pitch lag, L, is represented by seven bits and ranges between 17 and 143 inclusive.[12] The pitch gain, b, is represented by three bits and ranges from 0 to 2.0. For each pitch subframe, the speech coder determines and encodes the pitch lag, L, and the pitch gain b.

The method used to select the pitch parameters should be 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 pitch synthesis filter filtered by the LPC filter. The pitch synthesis uses the codebook vector with all zero elements as an input. The pitch lag, L, is selected from the set {17, 18, . . ., 143} and the pitch gain, b, is selected from the set {0, 0.25, 0.5, . . . , 2.0} (linearly quantized between 0 and 2.0 in steps of 0.25). The weighting filter is of the form:

$$W(z) = \frac{A(z)}{A(z/\zeta)}\ , \qquad \text{(A.1.4.5.1-2)}$$

where A(z) is the formant prediction error filter and ζ, which equals, 0.8, is a perceptual weighting parameter. The LPC coefficients used in the weighting filter are those for the current pitch subframe (see A.1.4.3.3.4 and A.1.4.3.3.5).

---

[12] L = 16 is reserved for the case when b = 0 (see A.1.4.5.1.3).

Reduced processing can be obtained by the filter arrangement shown in Figure A.1.4.5.1-1 (see Table A.1.4.5.1.1-1 for definitions of the symbols).

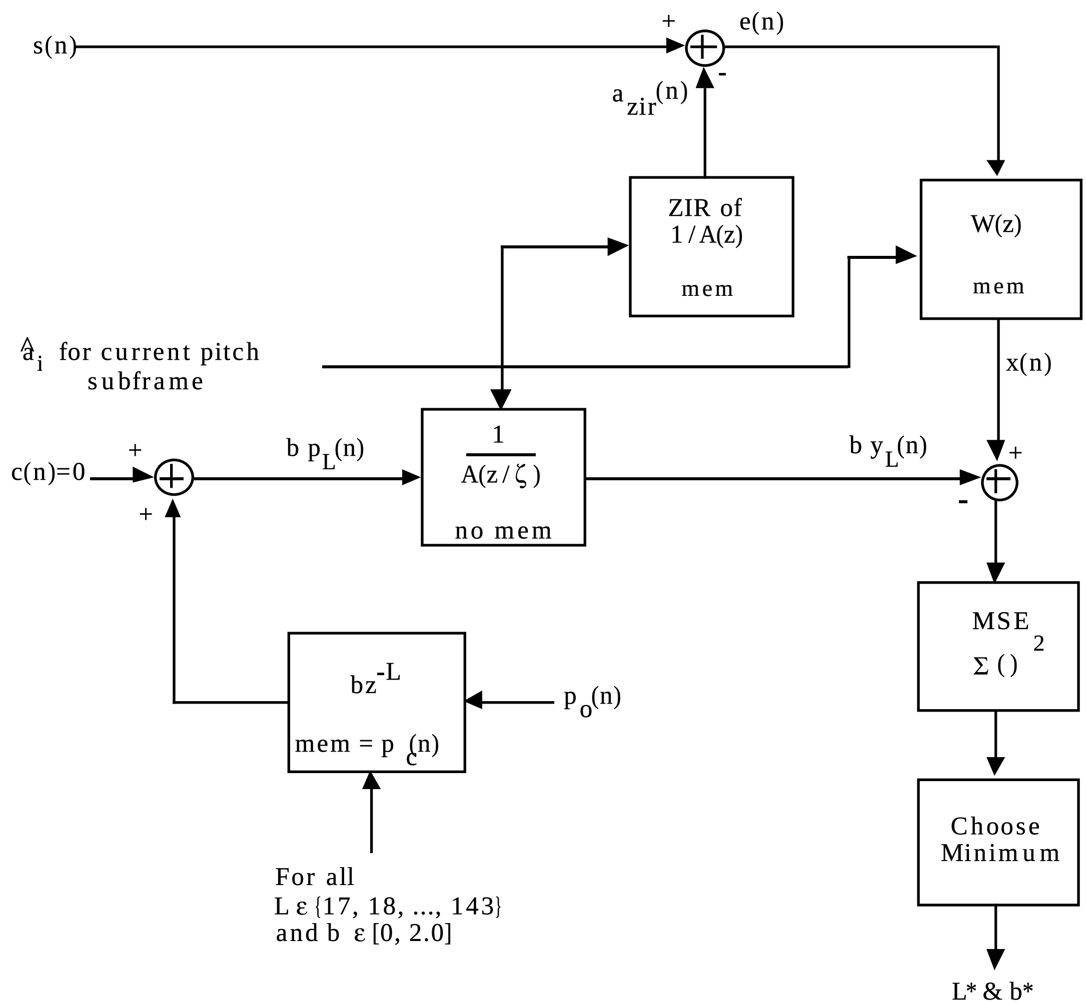

Figure A.1.4.5.1-1. Analysis-by-Synthesis Procedure for the Pitch Parameter Search

In this form, the synthesis filter used in the speech encoder is given by

$$H(z) = (1 / A(z))W(z) = \frac{1}{A(z/\zeta)}, \qquad \text{(A.1.4.5.1-3)}$$

which is the decoder speech synthesis filter followed by the perceptual weighting filter, and is therefore called the weighted synthesis filter.

A.1.4.5.1.1 Computing the Pitch Lag and Pitch Gain

The following terms are used to compute pitch lag and pitch gain:

Table A.1.4.5.1.1-1. Definition of Terms for Pitch Search

Term	Definition	Limits
$s(n)$	Input speech samples corresponding to the current pitch subframe with DC removed.	$0 \le n < L_P$
$a_{zir}(n)$	Zero input response of the formant 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 filter, the "closed loop formant residual." $p_c(-1)$ 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 output of the pitch filter, the open loop formant residual. This is $s(n)$ filtered by $A(z)$, using the appropriate LPC coefficients and memories (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 closed loop and open loop formant residual, 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 filter for lag L, with $b=1$.	$0 \le n < L_P$
$h(n)$	Impulse response of $H(z)$ which may be truncated to length of N_h elements (see A.1.4.5.1.2).	$0 \le n < N_h$
$y_L(n)$	$p_L(n)$ filtered by $H(z)$, assuming $H(z)$ has zero initial state.	$0 \le n < L_P$

Then define

$$E_{xyL} = \sum_{n=0}^{L_P-1} x(n) y_L(n) \qquad \text{(A.1.4.5.1.1-1)}$$

$$E_{yyL} = \sum_{n=0}^{L_P-1} y_L^2(n) \quad . \tag{A.1.4.5.1.1-2}$$

The optimal L, denoted by L*, and the optimal b, denoted by b*, are those values of L and b that result in the minimal mean square error of:

$$\sum_{n=0}^{L_P-1} \left\{ x(n) - b y_L(n) \right\}^2 \quad . \tag{A.1.4.5.1.1-3}$$

This minimum may be computed by searching for the minimum of

$$-2\ b\ E_{xyL} + b^2\ E_{yyL} \tag{A.1.4.5.1.1-4}$$

over the space of L and the eight quantized values of b. The allowable quantized values are discussed in A.1.4.5.1.

A.1.4.5.1.2 Implementing the Pitch Search Convolutions

Note that

$$y_L(n) = h(n) * p_L(n), \qquad 16 < L \le 143, \qquad 0 \le n < L_P,$$

$$= \sum_{i=0}^{n} h(i) p_L(n-i) \qquad 16 < L \le 143\ , \qquad 0 \le n < L_P. \tag{A.1.4.5.1.2-1}$$

The convolution can be truncated because the impulse response of the weighted synthesis filter, h(n), is typically small for n>20. Setting N_h equal to 20, Equation A.1.4.5.1.2-1 is approximated by

$$y_L(n) = \sum_{i=0}^{\min(n,N_h-1)} h(i) p_L(n-i)\ , \qquad 16 < L \le 143, \qquad 0 \le n < L_P. \tag{A.1.4.5.1.2-2}$$

Note also that

$$p_L(n) = p(n - L) = p_{L-1}(n - 1), \qquad 16 < L \le 143, \qquad 0 \le n < L_P. \tag{A.1.4.5.1.2-3}$$

From Equation A.1.4.5.1.2-2 and Equation A.1.4.5.1.2-3,

$$y_L(n) = \begin{cases} h(0)p(-L) & n=0 \\ y_{L-1}(n-1) + h(n)p(-L) & 1 \le n < N_h \\ y_{L-1}(n-1) & N_h \le n < L_P \end{cases} \qquad 17 < L \le 143, \tag{A.1.4.5.1.2-4}$$

In this way, once the initial convolution for $y_{17}(n)$ is done, the remaining convolutions can be done recursively by Equation A.1.4.5.1.2-4.

A.1.4.5.1.3 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. 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. When b* equals 0, the pitch lag is unnecessary as $1/P(z) = 1$ for all L. Because of this, L equal to 16 is used to represent the case when b* equals 0.

The chosen lag, L*, is an integer from 17 to 143. The resulting 128 possible values are coded using seven bits as follows: If b* equals 0, then PGAIN and PLAG are both set to 0. Otherwise, PGAIN is set to b*/0.25 - 1 and PLAG is set to L* -16.

A.1.4.5.2 Decoding

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

A.1.4.6 Determining the Codebook Parameters

A.1.4.6.1 Encoding

Except for a Rate 1/8 packet, each pitch subframe consists of two codebook subframes. For each codebook subframe, the speech coder determines the codebook index, I, and the codebook gain, G. For a Rate 1/8 packet, only one codebook index and one codebook gain is determined for each frame and the index is discarded before transmission (see A.1.4.6.1.3.2).

The excitation codebook vocoder codebook consists of 2^M code vectors, where M = 7. The codebook is organized in a recursive fashion such that each code vector differs from the adjacent code vector by one sample. The samples in adjacent code vectors are shifted by one position such that a new sample is shifted in at one end and a sample is dropped at the other (see the definition of $c_I(n)$ in Table A.1.4.6.1.1-1). Therefore a recursive codebook can be stored as a linear array that is $2^M + L_C - 1$ samples long. However, to simplify the implementation and to conserve memory space, a circular codebook 2^M samples long (128 samples) is used.

The codebook consists of the 128 code vectors having the values given in Table A.1.4.6.1-1. The values are in signed decimal notation. The table reads left to right, top to bottom, such that c(1) equals -2 and c(3) equals -1.5.

Table A.1.4.6.1-1. Codebook

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
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
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
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
-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
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
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
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

The method used to select the codebook vector and gain should be an analysis-by-synthesis method similar to that used for the pitch parameters search procedure. The chosen codebook index, I, and the codebook gain, G, are the values which minimize the weighted error between the synthesized speech and the input speech. The synthesized speech is the output of the codebook generator filtered by the pitch synthesis filter and the LPC filter. The weighting filter is of the form:

$$W(z) = \frac{A(z)}{A(z/\zeta)} , \qquad \text{(A.1.4.6.1-1)}$$

where A(z) is the formant prediction error filter and $\zeta = 0.8$ is a perceptual weighting parameter. The LPC coefficients used in the weighting filter are those resulting from the interpolation of the LSPs and LSP to LPC computation for the current codebook subframe (see A.1.4.3.3.4 and A.1.4.3.3.5).

Reduced processing can be obtained by the filter arrangement shown in Figure A.1.4.6.1-1.

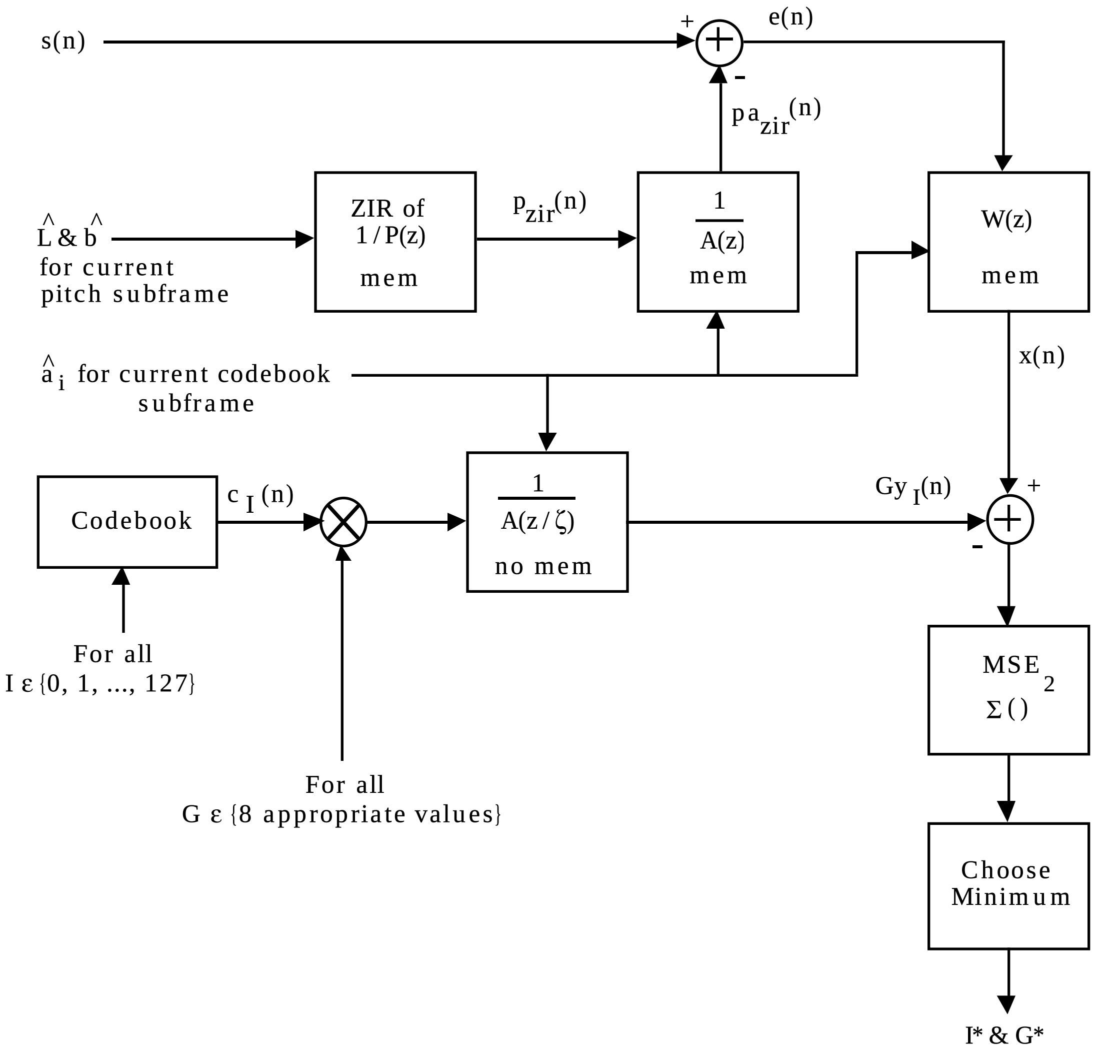

Figure A.1.4.6.1-1. Analysis-by-Synthesis Procedure for Codebook Parameter Search

A.1.4.6.1.1 Computing the Codebook Index and Codebook Gain

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

Table A.1.4.6.1.1-1. Definition of Terms for Codebook Search

Term	**Definition**	**Limits**
$s(n)$	Input speech samples corresponding to the current codebook subframe.	$0 \le n < L_C$
$p_{zir}(n)$	Zero input response of the pitch filter, with $\hat{L}$ and $\hat{b}$ 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.	$0 \le n < L_C$
$c(n)$	Random Gaussian center clipped codebook.	$0 \le n < 128$
$c_I(n)$	$c((n-I) \text{ modulo } 128)$. The output of the codebook for index I.	$0 \le n < L_C$
$h(n)$	Impulse response of $H(z)$ truncated to N_h samples (see A.1.4.5.1.2).	$0 \le n < N_h$
$y_I(n)$	$c_I(n)$ filtered by $H(z)$, assuming $H(z)$ has an initial state of 0 in all memories. 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.	$0 \le n < L_C$

Now define:

$$E_{xyI} = \sum_{n=0}^{L_C-1} x(n) y_I(n) \qquad \text{(A.1.4.6.1.1-1)}$$

$$E_{yyI} = \sum_{n=0}^{L_C-1} y_I^2(n) \ . \qquad \text{(A.1.4.6.1.1-2)}$$

The optimal I, denoted by I*, and the optimal G, denoted by G*, are those values of I and G that result in the minimal mean square error of:

$$\sum_{n=0}^{L_C-1}\left\{ x(n)-Gy_I(n) \right\}^2 . \qquad \text{(A.1.4.6.1.1-3)}$$

This minimum is computed by searching for the minimum of

$$-2\ G\ E_{xyI} + G^2\ E_{yyI} \qquad \text{(A.1.4.6.1.1-4)}$$

over the space of I and the eight quantized values of G. The allowable quantized values are discussed in A.1.4.6.1.3.

A.1.4.6.1.2 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.

Note that:

$$y_I(n) = h(n) * c_I(n), \qquad 0 \le I < 128, \quad 0 \le n < L_C,$$

$$= \sum_{i=0}^{n} h(i)c_I(n-i)\ , \qquad 0 \le I < 128, \quad 0 \le n < L_C. \qquad \text{(A.1.4.6.1.2-1)}$$

Again, since h(n) is typically small for n > N_h, Equation A.1.4.6.1.2-1 is approximated by

$$y_I(n) = \sum_{i=0}^{\min(n,N_h-1)} h(i)c_I(n-i)\ , \qquad 0 \le I < 128, \quad 0 \le n < L_C. \qquad \text{(A.1.4.6.1.2-2)}$$

Note also that

$$c_I(n) = c(\ (n - I)\ \text{modulo}\ 128) = c_{I-1}(n - 1)\ , \quad 0 \le I < 128, \qquad 0 \le n < L_C.$$

$$\text{(A.1.4.6.1.2-3)}$$

From Equation A.1.4.6.1.2-2 and Equation A.1.4.6.1.2-3,

$$y_I(n) = \begin{cases} h(0)c((-I)\text{modulo}128) & n=0 \quad 0 \le I < 128 \\ y_{I-1}(n-1)+h(n)c((-I)\text{modulo}128) & 1 \le n < N_h \quad 1 \le I < 128 \\ y_{I-1}(n-1) & N_h \le n < L_C \quad 1 \le I < 128 \end{cases} \qquad \text{(A.1.4.6.1.2-4)}$$

Once the initial convolution for $y_0(n)$ is completed, the remaining convolutions can be done recursively by Equation A.1.4.6.1.2-4. Note also that when c((-I)modulo 128) = 0,

$$y_I(n) = \begin{cases} 0 & n=0 \quad 1 \le I < 128 \\ y_{I-1}(n-1) & 1 \le n < L_C \quad 1 \le I < 128 \end{cases} \qquad \text{(A.1.4.6.1.2-5)}$$

A.1.4.6.1.3 Converting Codebook Parameters into Transmission Codes

A.1.4.6.1.3.1 Converting Codebook Parameters for All Rates Except 1/8

Figure A.1.4.6.1.3.1-1 shows the conversion scheme used for all rates except Rate 1/8.

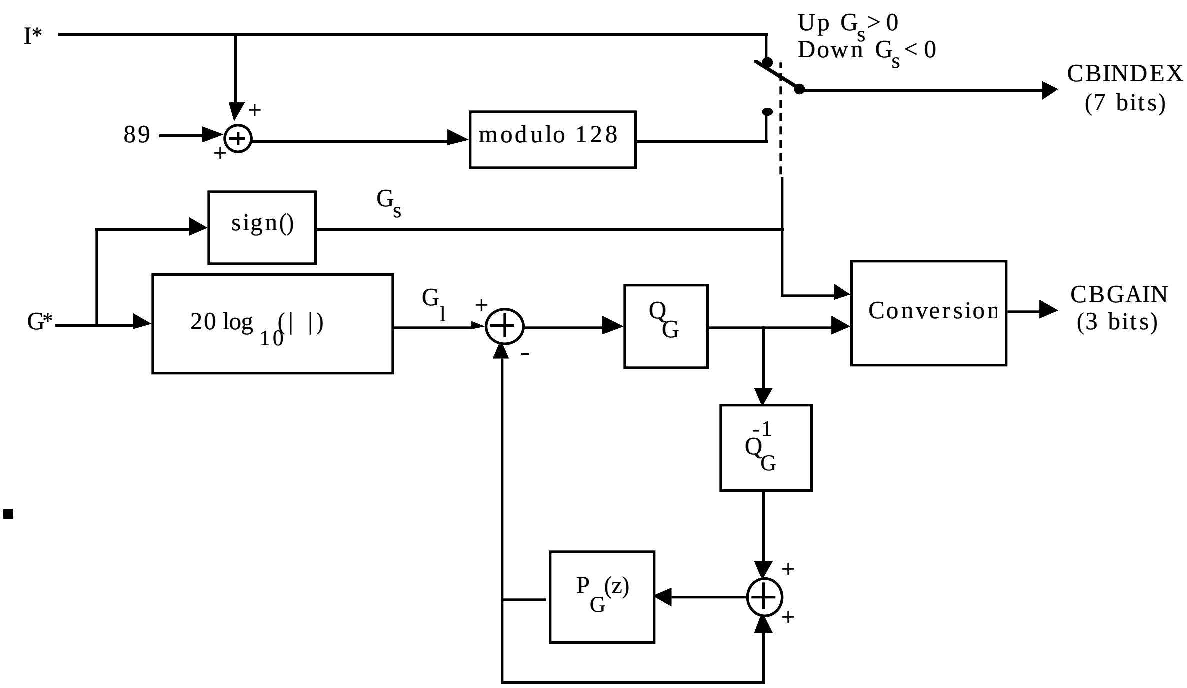

Figure A.1.4.6.1.3.1-1. Converting Codebook Parameters for All Rates Except Rate 1/8

One bit is used to represent the sign of the codebook gain, G_S, where G_S is defined as:

$$G_S = \mathrm{sign}(G^*), \qquad \text{(A.1.4.6.1.3.1-1)}$$

where

$$\mathrm{sign}\ (x) = \begin{cases} 1 & x \ge 0 \\ -1 & x < 0 \end{cases} \quad . \qquad \text{(A.1.4.6.1.3.1-2)}$$

The magnitude of the codebook gain is coded using a single differential coder operating on the log of the magnitude of G, as follows:

$$G_l = 20 \log_{10}(|G^*|). \qquad \text{(A.1.4.6.1.3.1-3)}$$

The differential coder employs a 2-bit linear quantizer Q_G and a codebook gain prediction filter $P_G(z)$. This differential coder operates on a codebook subframe basis regardless of the rate chosen for the frame. That is, the differential coder operates eight times during a Rate 1 frame, four times during a Rate 1/2 frame, two times during a Rate 1/4 frame, and only once during an Rate 1/8 frame. The predictor $P_G(z)$ is defined as

$$P_G(z) = F_G\left(\left\lfloor \frac{z^{-1}+z^{-2}}{2} \right\rfloor\right) \qquad \text{(A.1.4.6.1.3.1-4)}$$

where $\lfloor x \rfloor$ is the largest integer less than or equal to x, and $F_G(x)$ is defined in Table A.1.4.6.1.3.1-1.

Table A.1.4.6.1.3.1-1. Codebook Gain Prediction Filter Function $F_G(x)$

x	$F_G(x)$
-6	-2
-5	-2
-4	-2
-3	-2
-2	-1
-1	0
0	0
1	0
2	1
3	2
4	3
5	4
6	5
7	6
8	7
9	8
10	9
11	10
12	11
13	12
14	13
15	14
16	15
17	16
18	17
19	18
20	18
21	18
22	19
23	20
24	21
25	22
26	23
27	24
28	25
29	26
30	27
31	27
32	28
33	29
34	30
35	31
36	32
37	33
38	34
39	35
40	36
41	36
42	37
43	38
44	39
45	40
46	41
47	42
48	43
49	44
50	45
51	45
52	46
53	47
54	48
55	49
56	50
57	51
58	52
59	53
60	54
61	54
62	55
63	56
64	57
65	58
66	58

The difference between the current G_l and the output of $P_G(z)$ is then linearly quantized by $Q_G(x)$ as shown in Table A.1.4.6.1.3.1-2 and Table A.1.4.6.1.3.1-3.

Table A.1.4.6.1.3.1-2. Codebook Quantizer (Rate 1 and Rate 1/2)

Range of x	$Q_G(x)$
x < -2	-4
-2 ≤ x < 2	0
2 ≤ x < 6	4
6 ≤ x	8

Table A.1.4.6.1.3.1-3. Codebook Quantizer (Rate 1/4 and Rate 1/8)

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

The output of the quantizer, $Q_G(x)$, and the sign, G_s, is converted to CBGAIN as shown in Tables A.1.4.6.1.3.1-4 and A.1.4.6.1.3.1-5.

Table A.1.4.6.1.3.1-4. Conversion Table for CBGAIN (Rate 1 and Rate 1/2)

G_S	$Q_G(x)$	CBGAIN
+1	-4	0
+1	0	1
+1	4	2
+1	8	3
-1	-4	4
-1	0	5
-1	4	6
-1	8	7

Table A.1.4.6.1.3.1-5. Conversion Table for CBGAIN (Rate 1/4 and Rate 1/8)

G_S	$Q_G(x)$	CBGAIN
+1	-4	0
+1	-2	1
+1	0	2
+1	2	3
-1	-4	4
-1	-2	5
-1	0	6
-1	2	7

If G_S is negative, CBINDEX is set equal to (I+89) modulo 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.

A.1.4.6.1.3.2 Converting Codebook Parameters for Rate 1/8

The conversion scheme shown in Figure A.1.4.6.1.3.2-1 is used only for Rate 1/8.

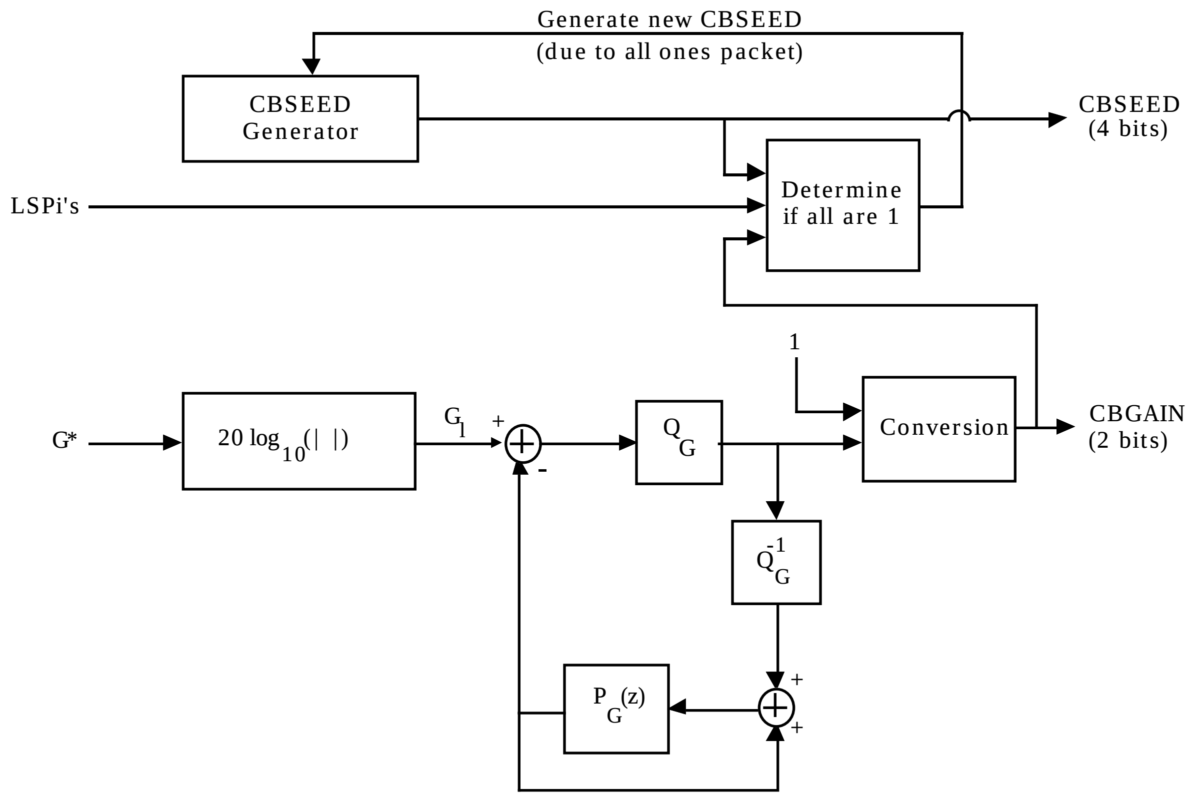

Figure A.1.4.6.1.3.2-1. Converting Codebook Parameters for Rate 1/8

For Rate 1/8 frames, the center-clipped Gaussian random codebook 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 magnitude of the codebook gain is quantized for transmission in exactly the same way as described above, with the exception that G_S is always +1, resulting in a 2-bit CBGAIN value between 0 and 3.

The pseudorandom code vector is generated by a pseudorandom number generator that is the same in the decoding sections of the transmitting encoder and the receiving decoder. This is accomplished by using the transmitted 16-bit data packet at Rate 1/8 as the seed for the pseudorandom number generator at both ends of transmission (see A.1.4.8.1.2).

To ensure the randomness of the transmitted packet, four pseudorandom bits are put into CBSEED. 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 which has been found to have satisfactory properties is

$$SD(new) = (521\,(SD(old)\,) + 259) \bmod 2^{16}\,. \qquad \text{(A.1.4.6.1.3.2-1)}$$

For each new transmitted Rate 1/8 packet, a new SD is computed and the four bits of CBSEED are given by

$$CBSEED[k] = SD(new)\,[\,4k + 3] \quad k = 0, 1, 2, 3 \qquad \text{(A.1.4.6.1.3.2-2)}$$

where CBSEED[k] denotes bit k of CBSEED. SD(new) is then saved for use as SD(old) in the next Rate 1/8 packet. At transmitting encoder initialization SD is set to 0.

As an example, if the previous value of SD = 40481 then

$$\begin{aligned} SD(new) &= (521(40481) + 259) \bmod 2^{16} \qquad \text{(A.1.4.6.1.3.2-1)} \\ &= 53804. \end{aligned}$$

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

A 1200 bps frame consisting of all ones is null Traffic Channel data. The vocoder does not supply a Rate 1/8 packet with all ones bits to the multiplex sublayer. If an all ones Rate 1/8 packet occurs after packing (see A.1.4.7.4), 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.

A.1.4.6.2 Decoding

A.1.4.6.2.1 Converting Codebook Transmission Codes for All Rates Except 1/8

Decoding of the codebook parameters is done by the reverse of the transformation described above. This is shown in Figure A.1.4.6.2.1-1.

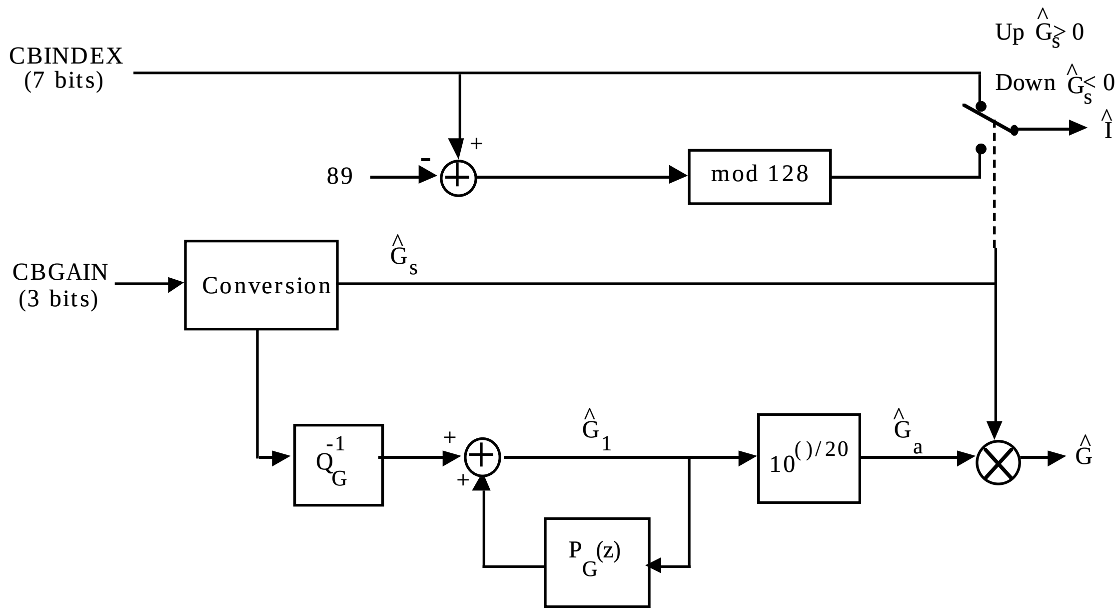

Figure A.1.4.6.2.1-1. Converting Codebook Transmission Codes for All Rates Except 1/8

The sign of the codebook gain $\hat{G}_s$ is set to +1 if CBGAIN is less than 4 and -1 if CBGAIN is greater than or equal to 4. For Rate 1 and Rate 1/2, the least significant two bits of CBGAIN are converted back into -4, 0, 4, or 8 as shown in Table A.1.4.6.1.3.1-4. For Rate 1/4, the least significant two bits of CBGAIN are converted back into -4, -2, 0, or 2 as shown in Table A.1.4.6.1.3.1-5. This value is then added to the predictor $P_G(z)$ to obtain the decoded value of $\hat{G}_l$.

The decoded $\hat{G}_l$ is then converted back into the linear domain via Table A.1.4.6.2.1-1. The values in this table correspond to the linear values of $\hat{G}_a$ with three fractional bits. Finally, $\hat{G}$ is found by multiplying $\hat{G}_a$ by $\hat{G}_s$.

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

Table A.1.4.6.2.1-1. Conversion Table for $\hat{G}_l$ to $\hat{G}_a$

$\hat{G}_l$	$\hat{G}_a$	$\hat{G}_l$	$\hat{G}_a$	$\hat{G}_l$	$\hat{G}_a$	$\hat{G}_l$	$\hat{G}_a$
-6	0.500	14	5.000	34	50.125	54	501.125
-5	0.500	15	5.625	35	56.250	55	562.375
-4	0.625	16	6.250	36	63.125	56	631.000
-3	0.750	17	7.125	37	70.750	57	708.000
-2	0.750	18	8.000	38	79.375	58	794.375
-1	0.875	19	8.875	39	89.125	59	891.250
0	1.000	20	10.000	40	100.000	60	1000.000
1	1.125	21	11.250	41	112.250	61	1122.000
2	1.250	22	12.625	42	125.875	62	1258.875
3	1.375	23	14.125	43	141.250	63	1412.500
4	1.625	24	15.875	44	158.500	64	1584.875
5	1.750	25	17.750	45	177.875	65	1778.250
6	2.000	26	20.000	46	199.500	66	1995.250
7	2.250	27	22.375	47	223.875		
8	2.500	28	25.125	48	251.250		
9	2.875	29	28.125	49	281.875		
10	3.125	30	31.625	50	316.250		
11	3.500	31	35.500	51	354.875		
12	4.000	32	39.750	52	398.125		
13	4.500	33	44.625	53	446.625		

A.1.4.6.2.2 Converting Codebook Transmission Codes for Rate 1/8

The procedure for determining the gain is shown in Figure A.1.4.6.2.2-1. The least significant two bits of CBGAIN are converted back into -4, -2, 0, or 2 as shown in Table A.1.4.6.1.3.1-5. 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 A.1.4.8.1.2).

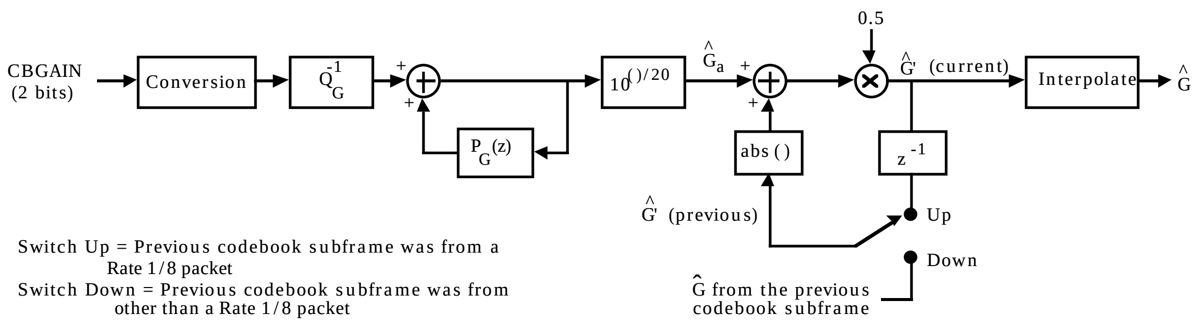

Figure A.1.4.6.2.2-1. Converting Codebook Transmission Codes for Rate 1/8

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

$$\hat{G}'(\text{current}) = 0.5\ |\hat{G}'(\text{previous})| + 0.5\ \hat{G}_a(\text{current}), \qquad \text{(A.1.4.6.2.2-1)}$$

where $\hat{G}_a$ (current) is the decoded linear codebook gain for the current codebook subframe, $\hat{G}'$ (previous) is the filtered linear codebook gain for the previous codebook 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). Since $\hat{G}_a$ (current) is guaranteed to be positive, the absolute value of $\hat{G}_a$ (current) does not need to be taken.

The value of $\hat{G}'$ is then interpolated to produce a smoother-sounding background noise:

$$\hat{G} = \begin{cases} 0.875\ \hat{G}'\ (\text{previous}) + 0.125\ \hat{G}'\ (\text{current}) & 0 \le n < 20 \\ 0.750\ \hat{G}'\ (\text{previous}) + 0.250\ \hat{G}'\ (\text{current}) & 20 \le n < 40 \\ 0.625\ \hat{G}'\ (\text{previous}) + 0.375\ \hat{G}'\ (\text{current}) & 40 \le n < 60 \\ 0.500\ \hat{G}'\ (\text{previous}) + 0.500\ \hat{G}'\ (\text{current}) & 60 \le n < 80 \\ 0.375\ \hat{G}'\ (\text{previous}) + 0.625\ \hat{G}'\ (\text{current}) & 80 \le n < 100 \\ 0.250\ \hat{G}'\ (\text{previous}) + 0.750\ \hat{G}'\ (\text{current}) & 100 \le n < 120 \\ 0.125\ \hat{G}'\ (\text{previous}) + 0.875\ \hat{G}'\ (\text{current}) & 120 \le n < 140 \\ \hat{G}'\ (\text{current}) & 140 \le n < 160 \end{cases} \qquad (A.1.4.6.2.2\text{-}2)$$

A.1.4.7 Data Packing

A.1.4.7.1 Rate 1 Parity Check Bits and Packing

A.1.4.7.1.1 Parity Check Bits

Eleven parity check bits are added to provide error correction and detection of the 18 most perceptually significant bits of the Rate 1 data. The error protection uses a cyclic code to generate ten parity check bits to form a (28, 18) code.[13] Then a single parity check bit is computed using the 28 bits of this code. This forms the final (29, 18) code.

The 18 most-significant bits are assembled into an input polynomial in GF(2) as follows:[14]

$$\begin{aligned} a(x) = {} & LSP1[3]\ x^{17} + LSP2[3]\ x^{16} + LSP3[3]\ x^{15} + LSP4[3]\ x^{14} \\ & + LSP5[3]\ x^{13} + LSP6[3]\ x^{12} + LSP7[3]\ x^{11} + LSP8[3]\ x^{10} \\ & + LSP9[3]\ x^{9} + LSP10[3]\ x^{8} + CBGAIN1[1]\ x^{7} + CBGAIN2[1]\ x^{6} \\ & + CBGAIN3[1]\ x^{5} + CBGAIN4[1]\ x^{4} + CBGAIN5[1]\ x^{3} \\ & + CBGAIN6[1]\ x^{2} + CBGAIN7[1]\ x^{1} + CBGAIN8[1]\ x^{0} \end{aligned} \qquad (A.1.4.7.1.1\text{-}1)$$

where LSPi[3] is the MSB of LSP code i, and CBGAINi[1] is the second MSB of CBGAIN code i. In effect, a(x) is made up of the MSBs of all ten LSP codes, and the second MSBs of the CBGAIN codes.

---

[13] The cyclic code is a shortened BCH code. The terminology (n, k) implies that the code word is n bits long and there are k information bits.

[14] GF(2) is the Galois Field of two elements. The multiplications and divisions are just ordinary multiplies and divides of one polynomial with another, except that the coefficients are restricted to be binary and the arithmetic is performed modulo 2. There are no carries or borrows. See Lin, S. and Costello, D. J., *Error Control Coding: Fundamentals and Applications,* (New Jersey: Prentice-Hall Inc., 1983), pp. 19-29.

The first ten parity check bits are found by using the cyclic code with generator polynomial of

$$g_{pc}(x) = x^{10} + x^9 + x^8 + x^6 + x^5 + x^3 + 1. \qquad \text{(A.1.4.7.1.1-2)}$$

r(x) is the remainder of the binary division of the input polynomial and the generator polynomial, or

$$a(x)\, x^{10} / g_{pc}(x) = q(x) + r(x)/g_{pc}(x)\ , \qquad \text{(A.1.4.7.1.1-3)}$$

where q(x) is the quotient of the division, and r(x) is the remainder of the division. The quotient is not used. The bits of r(x) are assigned as follows:[15]

$$\begin{aligned} r(x) = {} & \overline{\mathrm{PCB}[10]}\ x^9 + \overline{\mathrm{PCB}[9]}\ x^8 + \overline{\mathrm{PCB}[8]}\ x^7 + \overline{\mathrm{PCB}[7]}\ x^6 \\ & + \overline{\mathrm{PCB}[6]}\ x^5 + \overline{\mathrm{PCB}[5]}\ x^4 + \overline{\mathrm{PCB}[4]}\ x^3 \\ & + \overline{\mathrm{PCB}[3]}\ x^2 + \overline{\mathrm{PCB}[2]}\ x^1 + \overline{\mathrm{PCB}[1]}\ x^0\ . \qquad \text{(A.1.4.7.1.1-4)} \end{aligned}$$

The 11th protection bit, PCB[0], is a parity bit on the 18 protected bits in a(x) and the ten parity check bits in r(x). PCB[0] is ‘0’ if the exclusive-OR of all 28 bits results in ‘0’; PCB[0] is ‘1’ if the exclusive-OR of all 28 bits results in ‘1’. That is,

$$\begin{aligned} \mathrm{PCB}[0] = {} & \mathrm{LSP1}[3] \oplus \mathrm{LSP2}[3] \oplus \mathrm{LSP3}[3] \oplus \mathrm{LSP4}[3] \oplus \mathrm{LSP5}[3] \oplus \mathrm{LSP6}[3] \oplus \mathrm{LSP7}[3] \\ & \oplus \mathrm{LSP8}[3] \oplus \mathrm{LSP9}[3] \oplus \mathrm{LSP10}[3] \oplus \mathrm{CBGAIN1}[1] \oplus \mathrm{CBGAIN2}[1] \oplus \mathrm{CBGAIN3}[1] \\ & \oplus \mathrm{CBGAIN4}[1] \oplus \mathrm{CBGAIN5}[1] \oplus \mathrm{CBGAIN6}[1] \oplus \mathrm{CBGAIN7}[1] \oplus \mathrm{CBGAIN8}[1] \\ & \oplus \mathrm{PCB}[10] \oplus \mathrm{PCB}[9] \oplus \mathrm{PCB}[8] \oplus \mathrm{PCB}[7] \oplus \mathrm{PCB}[6] \oplus \mathrm{PCB}[5] \oplus \mathrm{PCB}[4] \\ & \oplus \mathrm{PCB}[3] \oplus \mathrm{PCB}[2] \oplus \mathrm{PCB}[1], \qquad \text{(A.1.4.7.1.1-5)} \end{aligned}$$

where $\oplus$ denotes the exclusive-OR of the operands.

[15]Note that PCB[1] through PCB[10] are inverted before transmission.

A.1.4.7.1.2 Rate 1 Packing

The 171 Rate 1 bits shall be packed into a primary traffic packet as shown in Table A.1.4.7.1.2-1. Bit 170 shall be the first primary traffic bit in the frame (it is just after the MM bit) and bit 0 shall be the last primary traffic bit in the frame (it is just before the first bit of the frame quality indicator).

Table A.1.4.7.1.2-1. Rate 1 Packet Structure (Part 1 of 2)

Bit	Code	Bit	Code	Bit	Code	Bit	Code
170	LSP1[2]	146	LSP3[1]	122	PLAG1[4]	98	CBGAIN2[2]
169	LSP1[3]	145	LSP3[0]	121	PLAG1[3]	97	CBGAIN2[0]
168	LSP2[2]	144	LSP4[1]	120	PLAG1[2]	96	PGAIN2[2]
167	LSP2[3]	143	CBGAIN1[1]	119	CBGAIN4[1]	95	CBGAIN7[1]
166	LSP3[2]	142	LSP4[0]	118	PLAG1[1]	94	PGAIN2[1]
165	LSP3[3]	141	LSP5[1]	117	PLAG1[0]	93	PGAIN2[0]
164	LSP4[2]	140	LSP5[0]	116	CBINDEX1[6]	92	PLAG2[6]
163	LSP4[3]	139	LSP6[1]	115	CBINDEX1[5]	91	PLAG2[5]
162	LSP5[2]	138	LSP6[0]	114	CBINDEX1[4]	90	PLAG2[4]
161	LSP5[3]	137	LSP7[1]	113	CBINDEX1[3]	89	PLAG2[3]
160	LSP6[2]	136	LSP7[0]	112	CBINDEX1[2]	88	PLAG2[2]
159	LSP6[3]	135	CBGAIN2[1]	111	CBGAIN5[1]	87	CBGAIN8[1]
158	LSP7[2]	134	LSP8[1]	110	CBINDEX1[1]	86	PLAG2[1]
157	LSP7[3]	133	LSP8[0]	109	CBINDEX1[0]	85	PLAG2[0]
156	LSP8[2]	132	LSP9[1]	108	CBGAIN1[2]	84	CBINDEX3[6]
155	LSP8[3]	131	LSP9[0]	107	CBGAIN1[0]	83	CBINDEX3[5]
154	LSP9[2]	130	LSP10[1]	106	CBINDEX2[6]	82	CBINDEX3[4]
153	LSP9[3]	129	LSP10[0]	105	CBINDEX2[5]	81	CBINDEX3[3]
152	LSP10[2]	128	PGAIN1[2]	104	CBINDEX2[4]	80	CBINDEX3[2]
151	LSP10[3]	127	CBGAIN3[1]	103	CBGAIN6[1]	79	PCB[10]
150	LSP1[1]	126	PGAIN1[1]	102	CBINDEX2[3]	78	CBINDEX3[1]
149	LSP1[0]	125	PGAIN1[0]	101	CBINDEX2[2]	77	CBINDEX3[0]
148	LSP2[1]	124	PLAG1[6]	100	CBINDEX2[1]	76	CBGAIN3[2]
147	LSP2[0]	123	PLAG1[5]	99	CBINDEX2[0]	75	CBGAIN3[0]

Table A.1.4.7.1.2-1. Rate 1 Packet Structure (Part 2 of 2)

Bit	Code
74	CBINDEX4[6]
73	CBINDEX4[5]
72	CBINDEX4[4]
71	PCB[9]
70	CBINDEX4[3]
69	CBINDEX4[2]
68	CBINDEX4[1]
67	CBINDEX4[0]
66	CBGAIN4[2]
65	CBGAIN4[0]
64	PGAIN3[2]
63	PCB[8]
62	PGAIN3[1]
61	PGAIN3[0]
60	PLAG3[6]
59	PLAG3[5]
58	PLAG3[4]
57	PLAG3[3]
56	PLAG3[2]
55	PCB[7]
54	PLAG3[1]
53	PLAG3[0]
52	CBINDEX5[6]
51	CBINDEX5[5]
50	CBINDEX5[4]
49	CBINDEX5[3]
48	CBINDEX5[2]
47	PCB[6]
46	CBINDEX5[1]
45	CBINDEX5[0]
44	CBGAIN5[2]
43	CBGAIN5[0]
42	CBINDEX6[6]
41	CBINDEX6[5]
40	CBINDEX6[4]
39	PCB[5]
38	CBINDEX6[3]
37	CBINDEX6[2]
36	CBINDEX6[1]
35	CBINDEX6[0]
34	CBGAIN6[2]
33	CBGAIN6[0]
32	PGAIN4[2]
31	PCB[4]
30	PGAIN4[1]
29	PGAIN4[0]
28	PLAG4[6]
27	PLAG4[5]
26	PLAG4[4]
25	PLAG4[3]
24	PLAG4[2]
23	PCB[3]
22	PLAG4[1]
21	PLAG4[0]
20	CBINDEX7[6]
19	CBINDEX7[5]
18	CBINDEX7[4]
17	CBINDEX7[3]
16	CBINDEX7[2]
15	PCB[2]
14	CBINDEX7[1]
13	CBINDEX7[0]
12	CBGAIN7[2]
11	CBGAIN7[0]
10	CBINDEX8[6]
9	CBINDEX8[5]
8	CBINDEX8[4]
7	PCB[1]
6	CBINDEX8[3]
5	CBINDEX8[2]
4	CBINDEX8[1]
3	CBINDEX8[0]
2	CBGAIN8[2]
1	CBGAIN8[0]
0	PCB[0]

A.1.4.7.2 Rate 1/2 Packing

The 80 Rate 1/2 bits shall be packed into a primary traffic packet as shown in Table A.1.4.7.2-1. Bit 79 shall be the first primary traffic bit in the frame and bit 0 shall be the last primary traffic bit in the frame.

Table A.1.4.7.2-1. Rate 1/2 Packet Structure

Bit	Code	Bit	Code	Bit	Code	Bit	Code
79	LSP1[1]	59	PGAIN1[2]	39	CBINDEX2[6]	19	CBINDEX3[6]
78	LSP1[0]	58	PGAIN1[1]	38	CBINDEX2[5]	18	CBINDEX3[5]
77	LSP2[1]	57	PGAIN1[0]	37	CBINDEX2[4]	17	CBINDEX3[4]
76	LSP2[0]	56	PLAG1[6]	36	CBINDEX2[3]	16	CBINDEX3[3]
75	LSP3[1]	55	PLAG1[5]	35	CBINDEX2[2]	15	CBINDEX3[2]
74	LSP3[0]	54	PLAG1[4]	34	CBINDEX2[1]	14	CBINDEX3[1]
73	LSP4[1]	53	PLAG1[3]	33	CBINDEX2[0]	13	CBINDEX3[0]
72	LSP4[0]	52	PLAG1[2]	32	CBGAIN2[2]	12	CBGAIN3[2]
71	LSP5[1]	51	PLAG1[1]	31	CBGAIN2[1]	11	CBGAIN3[1]
70	LSP5[0]	50	PLAG1[0]	30	CBGAIN2[0]	10	CBGAIN3[0]
69	LSP6[1]	49	CBINDEX1[6]	29	PGAIN2[2]	9	CBINDEX4[6]
68	LSP6[0]	48	CBINDEX1[5]	28	PGAIN2[1]	8	CBINDEX4[5]
67	LSP7[1]	47	CBINDEX1[4]	27	PGAIN2[0]	7	CBINDEX4[4]
66	LSP7[0]	46	CBINDEX1[3]	26	PLAG2[6]	6	CBINDEX4[3]
65	LSP8[1]	45	CBINDEX1[2]	25	PLAG2[5]	5	CBINDEX4[2]
64	LSP8[0]	44	CBINDEX1[1]	24	PLAG2[4]	4	CBINDEX4[1]
63	LSP9[1]	43	CBINDEX1[0]	23	PLAG2[3]	3	CBINDEX4[0]
62	LSP9[0]	42	CBGAIN1[2]	22	PLAG2[2]	2	CBGAIN4[2]
61	LSP10[1]	41	CBGAIN1[1]	21	PLAG2[1]	1	CBGAIN4[1]
60	LSP10[0]	40	CBGAIN1[0]	20	PLAG2[0]	0	CBGAIN4[0]

A.1.4.7.3 Rate 1/4 Packing

The 40 Rate 1/4 bits shall be packed into a primary traffic packet as shown in Table A.1.4.7.3-1. Bit 39 shall be the first primary traffic bit in the frame and bit 0 shall be the last primary traffic bit in the frame.

Table A.1.4.7.3-1. Rate 1/4 Packet Structure

Bit	Code	Bit	Code	Bit	Code	Bit	Code
39	LSP1[0]	29	PGAIN1[2]	19	CBINDEX1[6]	9	CBINDEX2[6]
38	LSP2[0]	28	PGAIN1[1]	18	CBINDEX1[5]	8	CBINDEX2[5]
37	LSP3[0]	27	PGAIN1[0]	17	CBINDEX1[4]	7	CBINDEX2[4]
36	LSP4[0]	26	PLAG1[6]	16	CBINDEX1[3]	6	CBINDEX2[3]
35	LSP5[0]	25	PLAG1[5]	15	CBINDEX1[2]	5	CBINDEX2[2]
34	LSP6[0]	24	PLAG1[4]	14	CBINDEX1[1]	4	CBINDEX2[1]
33	LSP7[0]	23	PLAG1[3]	13	CBINDEX1[0]	3	CBINDEX2[0]
32	LSP8[0]	22	PLAG1[2]	12	CBGAIN1[2]	2	CBGAIN2[2]
31	LSP9[0]	21	PLAG1[1]	11	CBGAIN1[1]	1	CBGAIN2[1]
30	LSP10[0]	20	PLAG1[0]	10	CBGAIN1[0]	0	CBGAIN2[0]

A.1.4.7.4 Rate 1/8 Packing

The 16 Rate 1/8 bits shall be packed into a primary traffic packet as shown in Table A.1.4.7.4-1. Bit 15 shall be the first primary traffic bit in the frame and bit 0 shall be the last primary traffic bit in the frame.

Table A.1.4.7.4-1. Rate 1/8 Packet Structure

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

A.1.4.8 Decoding at the Transmitting Vocoder and the Receiving Vocoder

At the encoder or transmit side, after each codebook subframe a version of the decoder (shown in Figure A.1.4.8-1) is run to update the filter memories used in the codebook searches. At the decoder or receive side, the decoder (shown in Figure A.1.4.8-2) decodes the received parameters to produce $s_d(n)$, the reconstructed speech. The two decoders are quite similar.

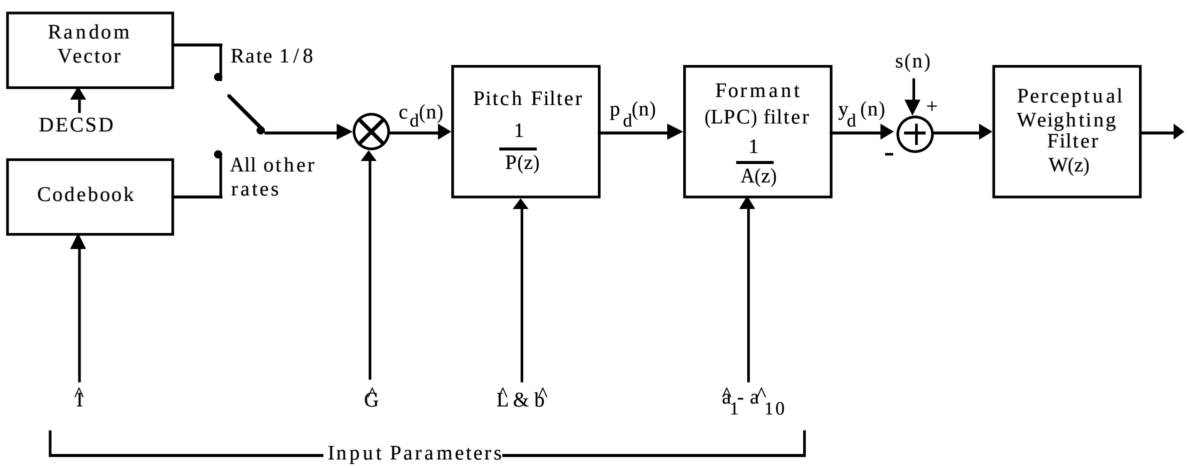

Figure A.1.4.8-1. Decoding at the Transmitting Vocoder

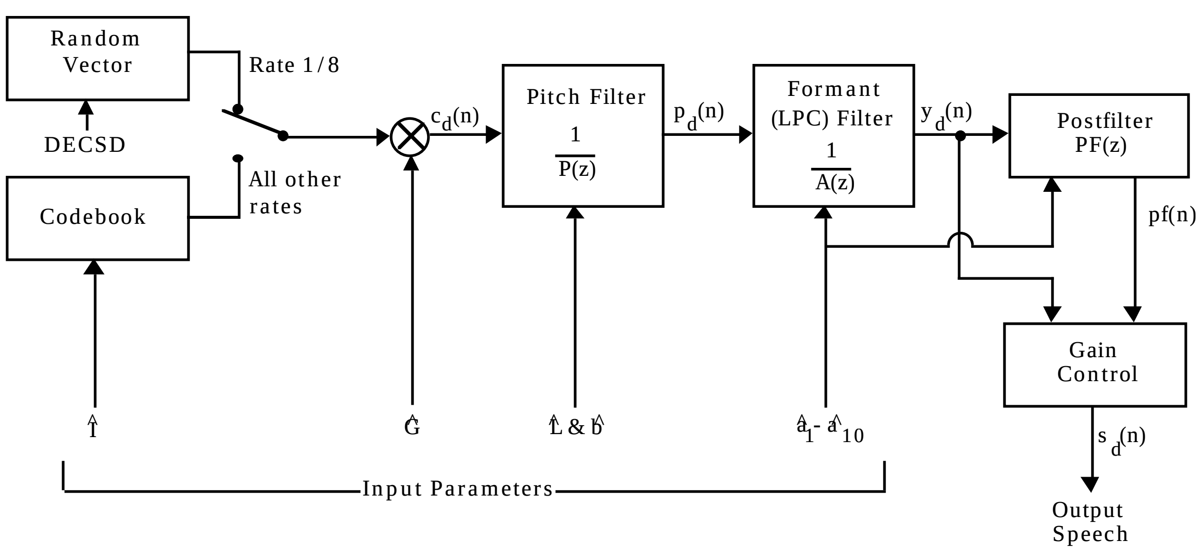

Figure A.1.4.8-2. Decoding at the Receiving Vocoder

A.1.4.8.1 Generating the Scaled Codebook Vector

Both the transmitting vocoder and the receiving vocoder generates the scaled codebook vector, $c_d(n)$. $c_d(n)$ is generated differently for a Rate 1/8 packet than for all other rate packets. $c_d(n)$ is generated in integer precision. All fractional bits are truncated. Fractional precision is used in computing $c_d(n)$; only the final result is truncated to integer precision.

A.1.4.8.1.1 Generating the Scaled Codebook Vector for All Rates Except Rate 1/8

First, $\hat{I}$ and $\hat{G}$ is decoded from CBGAIN and CBINDEX as previously described. $c_d(n)$ is then set to $\hat{G}$ c((n-$\hat{I}$) modulo 128), where c(n) is the random codebook in Table A.1.4.6.1-1.

A.1.4.8.1.2 Generating the Scaled Codebook Vector for Rate 1/8

For Rate 1/8 frames, $c_d(n)$ is set to a pseudorandom white sequence. Both the transmitting vocoder and the receiving vocoder 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.[16] DECSD is set to the 16-bit packet and is an integer. rnd(n), a length 160 random sequence having the same energy as the random center-clipped codebook, is then generated using the following procedure:[17]

```
{
  i=0
  decrv (old) = DECSD
  while (i < 160)
     {
       decrv (new) = (521 decrv (old) + 259) mod 2^16
       rnd (i) = (decrv (new) + 2^15) mod 2^16 - 2^15
       rnd (i) =  0.7931 √3 rnd (i)/32768.0
       decrv (old) = decrv (new)
       i = i + 1
     }
  }
```

The temporary variable decrv is an integer; the temporary variable rnd(n) is considered as a real number whose precision is described below. Intermediate integer calculations should be kept in full precision.

---

[16] 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.

[17] The sequence rnd(i) should have the same energy as the random codebook. A uniform probability density function from $\sqrt{3}$ to $-\sqrt{3}$ has variance 1, so rnd(i) is normalized to have variance 1 by multiplying the value by $\sqrt{3}$ /32768.0. However, the codebook has a variance of 0.629. To create a random sequence with variance of 0.629, rnd(i) is then multiplied by $\sqrt{0.629} = 0.7931$.

The resulting code vector, $c_d(n)$, is determined as follows:

$$c_d(n) = \hat{G}\,\mathrm{rnd}(n)\ , \tag{A.1.4.8.1.2-1}$$

where $\hat{G}$ is the interpolated gain value for the appropriate subframe (see A.1.4.6.2.2). Although $c_d(n)$ is computed without fractional bits, rnd(n) is computed with fractional precision, since rnd(n) is limited to be between $-\sqrt{1.887}$ and $+\sqrt{1.887}$. rnd(n) should be kept with at least 12 fractional bits, multiplied by the corresponding interpolated $\hat{G}$ value described above with three fractional bits, and then truncated to integer format.

A.1.4.8.2 Generating the Pitch Filter Output

Both the transmitting vocoder and the receiving vocoder generate the output of the pitch filter, $p_d(n)$, identically. The filter $1/P(z)$ is initialized with the final state resulting from the last sample of speech 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 $p_d(n)$. For Rate 1/8 frames, $\hat{b}$ is set to 0. The final state of the filter is saved for use in generating the speech for the next pitch subframe, and for use in the searches for the next pitch subframe in the encoder. The filter memories of $1/P(z)$, and $p_d(n)$ are kept in integer format, without fractional bits.

A.1.4.8.3 Generating the Formant Filter Output

Both the transmitting vocoder and the receiving vocoder generate the output of the formant filter, $y_d(n)$, identically. The LSP frequencies are interpolated appropriately for the codebook subframe of speech being generated, as described in A.1.4.3.3.4. The interpolated LSP frequencies are then converted back into LPC coefficients as described in A.1.4.3.3.5. The filter $1/A(z)$ is initialized with the final state resulting from the last sample of speech generated, but using $\hat{a}_i$ equal to the LPC coefficients generated for the current codebook subframe. $p_d(n)$ is filtered by $1/A(z)$ to produce $y_d(n)$. The final state of the filter is saved for use in generating the speech for the next codebook subframe, and for use in the searches for the next codebook subframe in the encoder. The filter memories of $1/A(z)$ and $y_d(n)$ are in integer format, without fractional bits.

A.1.4.8.4 Updating the Memories of W(z) in the Transmitting Vocoder

At the encoder, $s(n) - y_d(n)$ is then filtered by $W(z)$ to update the filter memories of $W(z)$ for use in the searches of the next codebook subframe of speech. The filter $W(z)$ is initialized with the final state resulting from the last sample of speech, but with $\hat{a}_i$ equal to the LPC coefficients appropriate for the current codebook subframe of speech. $s(n) - y_d(n)$ is filtered by $W(z)$. The output of this filter may be discarded. The final state of the filter is saved for use in the searches for the next codebook subframe.

A.1.4.8.5 The Adaptive Postfilter in the Receiving Vocoder

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

$$PF(z) = B(z)\,A(z/p)/A(z/s), \tag{A.1.4.8.5-1}$$

where A(z) is the formant prediction error filter defined in Equation A.1.4.3.1-1; p = 0.5 and s = 0.8. B(z) is an anti-tilt filter designed to offset the spectral tilt introduced by A(z/p)/A(z/s). B(z) is as follows:

$$B(z) = \frac{1-\gamma z^{-1}}{1+\gamma z^{-1}}, \qquad \text{(A.1.4.8.5-2)}$$

where γ is a function of the average of the ten interpolated LSP frequencies $\hat{w}'_i$ as follows:

$$\gamma = \begin{cases} 0.25 & \text{if average}(\hat{w}'_i) \le 0.24 \\ -25(\text{average}(\hat{w}'_i)-0.25) & \text{if } 0.24 < \text{average}(\hat{w}'_i) \le 0.26 \\ -0.25 & \text{if average}(\hat{w}'_i) > 0.26 \end{cases} \qquad \text{(A.1.4.8.5-3)}$$

Typically, the average of the ten LSP frequencies is less than 0.25, which results in a B(z) that is a high-pass brightener. Occasionally the average of the ten LSP frequencies is greater than 0.25, which results in a B(z) that is a low-pass dampener.

The filter PF(z) is initialized with the final state resulting from the last sample of speech. In this case, γ is a function of the current average of the ten interpolated LSP frequencies and the coefficients of A(z) are equal to the LPC coefficients appropriate for the current codebook subframe of speech (see A.1.4.3.2.4). $y_d(n)$ should then 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 roughly the same as the energy of the input signal. The input and output energies are computed on 40 sample intervals, regardless of the data rate selected. This is accomplished as follows:

First the energy of the input, $y_d(n)$, for the 40 samples of speech is computed as follows:

$$E_{in} = \sum_{n=0}^{39} y_d^2(n). \qquad \text{(A.1.4.8.5-4)}$$

The energy of the output, pf(n), is computed in the same manner:

$$E_{out} = \sum_{n=0}^{39} pf^2(n). \qquad \text{(A.1.4.8.5-5)}$$

An initial scale factor, $SCALE_{init}$, is computed as follows:

$$SCALE_{init} = \sqrt{\frac{E_{in}}{E_{out}}}. \qquad \text{(A.1.4.8.5-6)}$$

$SCALE_{init}$ is then filtered by a first order IIR filter to produce the final scale factor, $SCALE_{fin}$, by

$$SCALE_{fin}(\text{current}) = 0.9375\ SCALE_{fin}(\text{previous}) + 0.0625\ SCALE_{init}(\text{current}).$$

(A.1.4.8.5-7)

$SCALE_{init}$(current) is the $SCALE_{init}$ for the current 40 samples defined above and $SCALE_{fin}$(previous) is the $SCALE_{fin}$ from the previous 40 samples.

The reconstructed speech $s_d(n)$ is then computed as

$$s_d(n) = SCALE_{fin}(\text{current})\ pf(n). \qquad \text{(A.1.4.8.5-8)}$$

A.1.4.8.6 Special Cases

A.1.4.8.6.1 Insufficient Frame Quality (Erased) Packets

If the transmission rate cannot be satisfactorily determined, the multiplex sublayer informs the receiving vocoder of an erasure (see A.1.3.2.2). In addition, the receive vocoder may declare an erasure when it receives a Rate 1 packet and errors are detected (see A.1.4.8.6.2), when it receives an Rate 1 packet with probable bit errors and the number of errors exceeds one (see A.1.4.8.6.3), or when it receives a Rate 1/8 packet consisting of all ones (see A.1.4.8.6.5).

When the receive vocoder receives or declares an erased packet, the decoder decays all the parameters toward their initialization levels. The codebook gain for the entire frame is set to the previous codebook gain in dB, $\hat{G}_l$, multiplied by 0.7. The largest integer less than this value is then selected as the current value (in dB). This value is then entered into the predictor, so that in the next frame, the predictor will be a function of the average of the previous dB value and 0.7 times the previous dB value. The linear codebook gain, $\hat{G}_a$, is computed from Table A.1.4.6.2.1-1.[18] $\hat{G}_s$ is set equal to 1. In this way, multiple erasures will result in a steadily decreasing volume.

The codebook index is randomly chosen. The random codebook index is used as if the codebook subframe size is 160 samples.

The pitch filter should be effectively turned off by setting the pitch gain to zero.

The memories in the LSP predictors are multiplied by 0.90625, and LSP frequencies are regenerated from these memories. The LSP frequencies are checked for stability and are low-pass filtered using SM = 0.875 (see A.1.4.3.3.3). These uninterpolated LSP frequencies are then converted back into LPC coefficients, which are used for the entire frame of reconstructed speech.[19] In this way, multiple erasures will eventually lead to predictor

---

[18] Note that this is equivalent to multiplying the stored $\hat{G}_l$ by 0.7. The output of the predictor and inverse quantizer in Figure A.1.4.6.2.1-1 are ignored.

[19] Note that this is equivalent to having the output of Q_{wi}^{-1} in Figure A.1.4.3.3.1-1 equal to 0 for an erased packet.

memories equal to zero, resulting in LSP frequencies at their bias levels. This results in no shaping by the LPC filter, so erasures slowly move the LPC spectrum towards white noise.

These parameters are then used to reconstruct the current frame of speech and to update the filter memories for the next codebook subframe at the decoder.

A.1.4.8.6.2 Rate 1 Packets

The receiving vocoder evaluates a Rate 1 packet for bit errors. If the ten parity check bits from the cyclic code (bits PCB[1] through PCB[10]) show that the packet has no errors, the received frame is decoded as a normal Rate 1 frame. If the ten parity check bits from the cyclic code (bits PCB[1] through PCB[10]) show that the packet has an error, the packet is declared an erased packet and handled as described in A.1.4.8.6.1.

A.1.4.8.6.3 Rate 1 Packets with Probable Bit Errors

In certain cases, the decoder may receive a Rate 1 packet with probable bit errors. This received packet is most likely a full rate frame with errors. The decoder evaluates this data, and reconstructs the speech in different ways depending on the assessed quality of the received packet.

The decoder first verifies the eleven parity check bits. If the ten parity check bits from the cyclic code (bits PCB[1] through PCB[10]) show that the packet has no errors,[20] the received packet is decoded as a normal Rate 1 packet with the exception that the pitch filter is disabled ($\hat{b}$ is set to 0) for all four pitch subframes. The codebook parameters and LSP parameters are decoded and used as in a typical Rate 1 packet to reconstruct the speech and update the filter memories.

If the ten parity check bits from the cyclic code (bits PCB[1] through PCB[10]) show that the packet has only one bit in error and the parity bit (PCB[0]) does not check, the bit in error is corrected. Speech reconstruction is then completed as described in the previous paragraph.

If the ten parity check bits from the cyclic code (bits PCB[1] through PCB[10]) show that the packet has only one bit in error and the parity bit (PCB[0]) checks, or if the ten parity check bits (bits PCB[1] through PCB[10]) detect more than one bit in error, the frame is declared an erasure, and the speech is reconstructed as described in A.1.4.8.6.1.

A.1.4.8.6.4 Blanked Packets

A blanked frame occurs when the transmitting station uses the entire frame for either signaling traffic or secondary traffic. Blanking differs from erasing in that the encoder is aware that the packet is blanked, whereas the encoder is unaware of erasures. As such, the blanking algorithm is used at both the encoder and decoder for reconstructing the speech and updating the filter memories.

---

[20] See Lin and Costello, pp. 103-110 for a discussion of methods for determining whether there is an error and the number of errors.

For a blanked packet, the codebook gain is set to zero ($\hat{G}$ is set equal to 0), which effectively disables the codebook. However, changes are not made in the codebook predictor memories. The pitch parameters from the last pitch subframe of the previous frame are used, 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, which are used for the entire frame. However, changes are not made in the LSP predictor memories, $P_w(z)$.

From the above parameters, the speech is reconstructed and the filter memories are updated at both the encoder and the decoder.

A.1.4.8.6.5 All Ones Rate 1/8 Packets

A Rate 1/8 packet consisting of all ones is considered as null Traffic Channel data. This packet is declared an erased packet and handled as described in A.1.4.8.6.1.

A.1.4.9 Vocoder Initialization

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

- The filter and predictor memories are set to zero.
- The LSPs, $\hat{w}_i$(previous), are set to $Bias_i$ (see A.1.4.3.2.7 and A.1.4.3.3.1).
- The Rate 1/8 codebook gain, $\hat{G}'$ (old), is set to 0 (see A.1.4.8.1.2).
- The adaptive postfilter gain, $SCALE_{fin}$ (previous), is set to 1.0 (see A.1.4.8.5).

Upon being commanded to initialize the transmitting side, the vocoder 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 A.1.4.3.2.7 and A.1.4.3.3.1).
- The Rate 1/8 codebook gain, $\hat{G}'$ (old), is set to 0 (see A.1.4.8.1.2).
- The background noise level, B_1, is set to 160000, (see A.1.4.4.2).
- The Rate 1/8 random codebook seed, SD, is set to 0.

A.1.4.10 Output Audio Interface

A.1.4.10.1 Output Audio Interface in the Mobile Station

The output audio can be either an analog signal or an 8-bit µlaw PCM signal.

A.1.4.10.1.1 Digital Audio Output

If the output audio is an 8-bit µlaw PCM signal, it shall be converted from a uniform PCM format according Table 2 in CCITT Recommendation G.711.[21]

A.1.4.10.1.2 Analog Audio Output

If the output is in analog form, the mobile station converts the vocoder output samples to an analog speech signal. This may be done by the following method: the samples are first converted to a µlaw format, then to an analog signal, then band-pass filtered, and then adjusted to obtain the correct output level. Alternatively, the samples may be directly converted to analog or transformed by any other equivalent method.

A.1.4.10.1.2.1 Band Pass Filtering

Output reconstruction filtering shall conform to CCITT Recommendation G.714.[22] Additional reconstruction filtering may be provided by the manufacturer.

A.1.4.10.1.2.2 Receive Level Adjustment

Pending the generation of a complete speech transmission plan for dual-mode cellular systems, the following requirements shall be met to ensure compatibility with the transmission plan for fixed digital speech networks.

The mobile station shall have a nominal receive objective loudness rating (ROLR) equal to 51 dB when receiving from a reference base station (see A.1.4.2.2.2). The loudness ratings are described in IEEE Standard 661-1979.[23] Measurement techniques are described in "Recommended Minimum Performance Standards for 800 MHz Wideband Spread Spectrum Dual-Mode Mobile Stations."

A.1.4.10.2 Output Audio Interface in the Base Station

A.1.4.10.2.1 Digital Audio Output

A.1.4.10.2.1.1 Reserved

---

[21] See CCITT Recommendation "Pulse Code Modulation (PCM) of Voice Frequencies," Vol III, Recommendation G.711, Geneva 1972.

[22] See CCITT Recommendation "Separate Performance Characteristics for the Encoding and Decoding Sides of PCM Channels Applicable to 4-Wire Voice-Frequency Interfaces," Blue Book, Vol III, Recommendation G.714, Melbourne, 1988.

[23] See "IEEE Standard Method for Determining Objective Loudness Ratings of Telephone Connections," ANSI/IEEE Standard 661-1979.

A.1.4.10.2.1.2 Reserved

A.1.4.10.2.1.3 Receive Level Adjustment

Pending the generation of a complete speech transmission plan for dual-mode cellular systems, the following requirements shall be met to ensure compatibility with the transmission plan for fixed digital speech networks.

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 are described in “Recommended Minimum Performance Standards for 800 MHz Base Stations Supporting Wideband Spread Spectrum Dual-Mode Mobile Stations.”

A.1.4.11 Summary of Encoding and Decoding

A.1.4.11.1 Encoding Summary

The following summarizes the steps taken to encode a frame:

1.0 Initial Computations

1.1 Remove the DC from the current input speech.

1.2 Compute the LPC coefficients for the current frame.

1.3 Compute the LSP frequencies from the LPC coefficients.

1.4 Compute the data rate.

1.5 Convert the LSP frequencies into transmission codes.

1.6 If the packet is Rate 1/2, go to 3.0.

1.7 If the packet is Rate 1/4, go to 4.0.

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

1.9 If the packet is a blank packet, go to 6.0.

1.10 Go to 2.0.

2.0 Rate 1 Packet Encoding

2.1 Start with the first pitch subframe.

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

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

2.4 Find the optimal codebook gain and index for the first codebook subframe in the pitch subframe.

2.5 Update the pitch filter, formant filter, and perceptual weighting filter memories.

2.6 Find the optimal codebook gain and index for the second codebook subframe in the pitch subframe.

2.7 Update the pitch filter, formant filter, and perceptual weighting filter memories.

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

2.9 Compute the CRC and pack the data into the 171-bit packet.

2.10 Done encoding.

3.0 Rate 1/2 Packet Encoding

3.1 Start with the first pitch subframe.

3.2 Interpolate the LSPs for the pitch subframe and the two corresponding codebook subframes, then convert them back 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 first codebook subframe in the pitch subframe.

3.5 Update the pitch filter, formant filter, and perceptual weighting filter memories.

3.6 Find the optimal codebook gain and index for the second codebook subframe in the pitch subframe.

3.7 Update the pitch filter, formant filter, and perceptual weighting filter memories.

3.8 If both pitch subframes for this frame have not been done, go to the next pitch subframe and go to 3.2.

3.9 Pack the data into the 80-bit packet.

3.10 Done encoding.

4.0 Rate 1/4 Packet Encoding

4.1 Interpolate the LSPs for the pitch subframe and the two corresponding codebook subframes, then convert them back to LPC coefficients.

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

4.3 Find the optimal codebook gain and index for the first codebook subframe.

4.4 Update the pitch filter, formant filter, and perceptual weighting filter memories.

4.5 Find the optimal codebook gain and index for the second codebook subframe.

4.6 Update the pitch filter, formant filter, and perceptual weighting filter memories.

4.7 Pack the data into the 40-bit packet.

4.8 Done encoding.

5.0 Rate 1/8 Packet Encoding

5.1 Interpolate the LSPs for the codebook subframe, then convert them back to LPC coefficients.

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

5.3 Discard the index, generate CBSEED, and pack the data into the 16-bit packet.

5.4 Update the pitch, formant, and perceptual weighting filter memories using the 16-bit packet as the seed for the pseudo-random number generator in the decoder.

5.5 Done encoding.

6.0 Blank Packet Encoding

6.1 Set codebook gain to zero, without changing the predictor memories for the codebook gain.

6.2 Set the current pitch gain and lag to those of the last pitch subframe of the previous frame, and saturate the pitch gain to be no greater than 1.

6.3 Convert the previous frame's uninterpolated LSP frequencies to LPC coefficients, without changing the LSP predictor memories.

6.4 Update the pitch, formant, and perceptual weighting filter memories, using the codebook, pitch, and LPC parameters described above for the entire frame.

6.5 Done encoding.

A.1.4.11.2 Decoding Summary

The following summarizes the steps taken to decode a frame.

1.0 Initial Computations

1.1 If the received packet is Rate 1/2, go to 4.0.

1.2 If the received packet is Rate 1/4, go to 5.0.

1.3 If the received packet is Rate 1/8, go to 6.0.

1.4 If the received packet is blanked, go to 8.0.

1.5 If the received packet is of insufficient frame quality (erasure), go to 7.0.

1.6 If the received packet is Rate 1 with probable bit errors, go to 3.0.

1.7 Go to 2.0.

2.0 Rate 1 Packet Decoding

2.1 Unpack the 171-bit packet into the appropriate codes.

2.2 Check the internal packet parity check bits to determine if there are any detected errors. If there are any detected errors, then declare the packet has insufficient frame quality and go to 7.0.

2.3 Compute the vocoder parameters from the unpacked codes.

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

2.5 Compute the output of the pitch filter for all 160 samples from the scaled codebook vector computed above using the pitch lag and gain parameters for all four pitch subframes.

2.6 Interpolate the LSP frequencies four times (once for each pitch subframe) and convert these frequencies to the LPC coefficients used in the formant synthesis and adaptive postfilter for the four pitch subframes.

2.7 Compute the output of the formant filter for all 160 samples from the output of the pitch filter using the appropriate LPC coefficients for each pitch subframe of 40 samples.

2.8 Compute output of the adaptive postfilter and the reconstructed speech for all 160 samples from the output of the formant filter using the appropriate LPC coefficients for each pitch subframe of 40 samples.

2.9 Done decoding.

3.0 Rate 1 Packet with Probable Bit Errors Decoding

3.1 Check the internal packet parity check bits to see how many bit errors exist in the current packet.

3.2 If the number of bits in error is greater than one, or if there is one bit in error and the parity bit PCB[0] checks, go to 7.0.

3.3 If one bit is in error, correct it.

3.4 Unpack the 171-bit packet into the appropriate codes and compute the vocoder parameters from these codes.

3.5 Set b = 0 for all four pitch subframes.

3.6 Go to 2.4.

4.0 Rate 1/2 Packet Decoding

4.1 Unpack the 80-bit packet into the appropriate codes, and compute the vocoder parameters from these codes.

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

4.3 Compute output of the pitch filter for all 160 samples from the scaled codebook vector computed above using the pitch lag and gain parameters for both pitch subframes.

4.4 Interpolate the LSP frequencies two times (once for each pitch subframe) and convert these frequencies to the LPC coefficients used in formant synthesis and adaptive postfilter for the four codebook subframes.

4.5 Compute the output of the formant filter for all 160 samples from the output of the pitch filter using the appropriate LPC coefficients for each codebook subframe of 40 samples.

4.6 Compute the output of the adaptive postfilter and the reconstructed speech for all 160 samples from the output of the formant filter using the appropriate LPC coefficients for each codebook subframe of 40 samples.

4.7 Done decoding.

5.0 Rate 1/4 Packet Decoding

5.1 Unpack the 40-bit packet into the appropriate codes, and compute the vocoder parameters from these codes.

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

5.3 Compute the output of the pitch filter for all 160 samples from the scaled codebook vector computed above using the pitch lag and gain parameters for the pitch subframe.

5.4 Interpolate the LSP frequencies once (for the pitch subframe) and convert these frequencies to the LPC coefficients used in formant synthesis and adaptive postfilter for the two codebook subframes.

5.5 Compute output of the formant filter for all 160 samples from the output of the pitch filter using the appropriate LPC coefficients for each codebook subframe of 80 samples.

5.6 Compute the output of the adaptive postfilter and the reconstructed speech for all 160 samples from the output of the formant filter using the appropriate LPC coefficients for each codebook subframe of 80 samples.

5.7 Done decoding.

6.0 Rate 1/8 Packet Decoding

6.1 If the packet is all 1's (the frame was null Traffic Channel data), then go to 7.0.

6.2 Unpack the 16-bit packet into the appropriate codes and compute the vocoder parameters from these codes.

6.3 Compute the scaled codebook vector for all 160 samples using the 16-bit packet as the random seed for the pseudorandom number generator and low pass filtering and interpolating the codebook gain.

6.4 The output of the pitch filter equals the scaled codebook vector because the pitch filter is not used.

6.5 Interpolate the LSP frequencies once for the codebook subframe (the entire frame), and convert these frequencies to the LPC coefficients used in the formant synthesis and adaptive postfilter for the codebook subframes (the entire frame).

6.6 Compute the output of the formant filter for all 160 samples from the output of the pitch filter using the appropriate LPC coefficients for the codebook subframe of 160 samples.

6.7 Compute the output of the adaptive postfilter and the reconstructed speech for all 160 samples from the output of the formant filter again using the appropriate LPC coefficients for the codebook subframe of 160 samples.

6.8 Done decoding.

7.0 Insufficient Frame Quality (Erasure) Decoding

7.1 Decay the codebook gain magnitude in dB by 0.7, update the codebook gain magnitude predictor memories, and compute the resulting linear value of the codebook gain.

7.2 Select a random codebook index.

7.3 Compute the scaled codebook vector for all 160 samples using the codebook gain and index parameters generated above.

7.4 The output of the pitch filter equals the scaled codebook vector because the pitch filter is not used.

7.5 Decay the LSP predictor memories by 0.90625, compute the resulting LSP frequencies, and convert them into the LPC coefficients used in the formant synthesis and adaptive postfilter for the frame.

7.6 Compute the output of the formant filter for all 160 samples from the output of the pitch filter using the LPC coefficients computed above.

7.7 Compute the output of the adaptive postfilter and the reconstructed speech for all 160 samples from the output of the formant filter using the LPC coefficients computed above.

7.8 Done decoding.

8.0 Blank Packet Decoding

8.1 Set the codebook gain to zero, without changing the codebook gain predictor memories. The scaled codebook vector is the all zero vector.

8.2 Set the current pitch gain and lag to those of the last pitch subframe of the previous frame, and saturate the pitch gain to be at most unity.

8.3 Compute the output of the pitch filter for all 160 samples using the pitch gain and lag parameters defined above.

8.4 Set the current LSP frequenciesto be the uninterpolated LSP frequencies from the previous frame, without changing the LSP predictor memories. Compute the LPC coefficients from these LSP frequencies.

8.5 Compute the output of the formant filter for all 160 samples from the output of the pitch filter using the LPC coefficients computed above.

8.6. Compute the output of the adaptive postfilter and the reconstructed speech for all 160 samples from the output of the formant filter using the LPC coefficients computed above.

8.7 Done decoding.

A.1.4.12 Allowable Delays

A.1.4.12.1 Allowable Transmitting Vocoder Encoding Delay

The transmitting vocoder in the mobile station shall supply a packet to the multiplex sublayer not later than 20 ms after it has obtained the last input sample for the Hamming window (see A.1.4.3.2.2).

A.1.4.12.2 Allowable Receiving Vocoder Decoding Delay

The receiving decoder in the mobile station shall generate the first sample of speech using parameters from a packet supplied to it by the multiplex sublayer not later than 3 ms after being supplied the packet.

A.1.4.13 Summary of Service Option 1 Notation

Table A.1.4.13-1 lists the parameters used for Service Option 1, Variable Data Rate Two-Way Voice.

Table A.1.4.13-1. Summary of Service Option 1 Notation (Part 1 of 5)

Parameter	Section	Name/Description
α_i	A.1.4.3.2.5	Linear predictive coding coefficients before bandwidth expansion.
a(x)	A.1.4.7.1.1	Input polynomial in GF(2); used for parity checking.
a_i	A.1.4.3.2.5	Linear predictive coding coefficients.
$\hat{a}_i$	A.1.4.3.3.5	Quantized, smoothed and interpolated LPC coefficients.
$a_{zir}(n)$	A.1.4.5.1.1	Zero input response of the formant synthesis filter.
A(z)	A.1.4.3.1	Formant prediction error filter.
1/A(z)	A.1.4.3.1	Formant synthesis filter.
b	A.1.4.5.1	Pitch gain.
b*	A.1.4.5.1.1	Optimal pitch gain.
$\hat{b}$	A.1.4.5.2	Pitch gain used for synthesis.
β	A.1.4.3.2.5	Scaling factor for bandwidth expansion.
B_i	A.1.4.4.2	Background noise estimate for the ith frame.
$Bias_i$	A.1.4.3.2.7	Line spectral pair bias for LSP frequency i.
B(z)	A.1.4.8.5	Anti-tilt filter.
CBGAINi	A.1.4.1	Codebook gain for the ith codebook subframe.
CBINDEXi	A.1.4.1	Codebook index for the ith codebook subframe.
CBSEED	A.1.4.1	Four bit value to randomize Rate 1/8 packets.
$c_d(n)$	A.1.4.8.1	Scaled codebook vector.
$c_I(n)$	A.1.4.6.1.1	The codebook vector for index I.
c(n)	A.1.4.6.1.1	Random Gaussian center clipped codebook.
decrv	A.1.4.8.1.2	Random variable used in generating the Rate 1/8 code vector.
DECSD	A.1.4.8.1.2	The decoder seed for a Rate 1/8 packet. Equal to the entire packet.
e(n)	A.1.4.5.1.1 A.1.4.6.1.1	The error between the input speech signal and the response of the formant synthesis filter.
$E^{(i)}$	A.1.4.3.2.4	Energy of prediction error with LPC filter of order i.
E_{in}	A.1.4.8.5	Input energy to the adaptive postfilter.
E_{out}	A.1.4.8.5	Output energy of the adaptive postfilter.
E_{xyI}	A.1.4.6.1.1	The zero-offset cross correlation between the output of the perceptual weighting filter and the weighted synthesis filter for the codebook search.

Table A.1.4.13-1. Summary of Service Option 1 Notation (Part 2 of 5)

Parameter	Section	Name/Description
E_{yyI}	A.1.4.6.1.1	The energy output of the weighted synthesis filter for the codebook search.
E_{xyL}	A.1.4.5.1.1	The zero-offset cross correlation between the output of the perceptual weighting filter and the weighted synthesis filter for the pitch search.
E_{yyL}	A.1.4.5.1.1	The energy output of the weighted synthesis filter for the pitch search.
$F_G(x)$	A.1.4.6.1.3.1	Codebook gain prediction filter function.
γ	A.1.4.8.5	Anti-spectral tilt filter coefficient.
G	A.1.4.6.1	Codebook gain.
G*	A.1.4.6.1	Optimal codebook gain.
$\hat{G}$	A.1.4.6.2.1	Decoded codebook gain (Decoded, filtered and interpolated for Rate 1/8).
$\hat{G}_a$	A.1.4.6.2	Decoded linear codebook gain magnitude.
$\hat{G}'$	A.1.4.6.2.2	Decoded and filtered codebook gain (used for Rate 1/8 and erased packets).
G_l	A.1.4.6.1.3.1	Codebook gain magnitude in dB.
$\hat{G}_l$	A.1.4.6.2.1	Decoded codebook gain magnitude in dB.
$g_{pc}(x)$	A.1.4.7.1.1	Parity check generator polynomial.
G_s	A.1.4.6.1.3.1	Sign of the codebook gain.
$\hat{G}_s$	A.1.4.6.2.1	Sign of the decoded codebook gain.
h(n)	A.1.4.5.1.2	Impulse response of H(z).
H(z)	A.1.4.5.1	Weighted synthesis filter. The combined formant synthesis filter and perceptual weighting filter.
i	All sections	Index.
I	A.1.4.6.1	Codebook index.
I*	A.1.4.6.1	Index of optimal codeword.
$\hat{I}$	A.1.4.6.2.1	Codebook index used for synthesis.
$\alpha j^{(i)}$	A.1.4.3.2.4	LPC coefficient j of LPC filter of order i.
k	All sections	Index.
k_i	A.1.4.3.2.4	Partial correlation coefficients.
L	A.1.4.5.1	Pitch lag.

1 **Table A.1.4.13-1. Summary of Service Option 1 Notation (Part 3 of 5)**

Parameter	Section	Name/Description
L*	A.1.4.5.1.1	Optimal pitch lag.
$\hat{L}$	A.1.4.5.2	Pitch lag used for synthesis.
L_A	A.1.4.1	LPC frame length in samples.
L_C	A.1.4.1	Codebook subframe length in samples.
L_P	A.1.4.1	Pitch subframe length in samples.
LSPi	A.1.4.1	Transmission code for Line spectral pair frequency i.
N	A.1.4.3.2.7	Number of bits of quantization in $Q_{wi}(x)$.
N_h	A.1.4.5.1.1	Number of samples that are used from the impulse response of the weighted synthesis filter.
P	A.1.4.3.1	Order of linear predictive coding filter.
$P_A(z)$	A.1.4.3.2.6	Intermediate polynomial used in transforming the LPC coefficients to LSP frequencies.
$\hat{P}_A(z)$	A.1.4.3.3.5	Intermediate polynomial used in transforming the interpolated LSP frequencies to LPC coefficients.
$pa_{zir}(n)$	A.1.4.6.1.1	Zero input response of the cascade of the pitch and formant synthesis filters.
PCB	A.1.4.1	Parity check bits for Rate 1 packets.
$p_c(n)$	A.1.4.5.1.1	Past outputs of the pitch filter.
$p_d(n)$	A.1.4.8.2	Output of the pitch filter.
PF(z)	A.1.4.8.5	Adaptive post filter.
pf(n)	A.1.4.8.5	Output of the adaptive post filter.
$P_G(z)$	A.1.4.6.1.3.1	Codebook gain prediction filter.
PGAINi	A.1.4.1	Transmission code for the pitch gain for the ith pitch subframe.
p_i	A.1.4.3.2.6	Coefficients of $P_A(z)$.
$\hat{p}_i$	A.1.4.3.3.5	Coefficients of $\hat{P}_A(z)$.
p'_i	A.1.4.3.2.6	Coefficients of P'(w).
$p_L(n)$	A.1.4.5.1.1	Estimated output of the pitch filter for lag L with b = 1.
PLAGi	A.1.4.1	Pitch lag for the ith pitch subframe.
p(n)	A.1.4.5.1.1	Combined closed loop and open loop formant residual.
$p_o(n)$	A.1.4.5.1.1	Estimate of the future output of the pitch filter.

1 **Table A.1.4.13-1. Summary of Service Option 1 Notation (Part 4 of 5)**

Parameter	**Section**	**Name/Description**
$P'(w)$	A.1.4.3.2.6	Function used in computing LSP frequencies.
$P_W(z)$	A.1.4.3.2.7	Prediction filter used in converting LSP frequencies.
$1/P(z)$	A.1.4.1	Pitch synthesis filter.
$p_{zir}(n)$	A.1.4.6.1.1	Zero input response of the pitch filter.
$Q_A(z)$	A.1.4.3.2.6	Intermediate polynomial used in transforming the LPC coefficients to LSP frequencies.
$\hat{Q}_A(z)$	A.1.4.3.3.5	Intermediate polynomial used in transforming the interpolated LSP frequencies to LPC coefficients.
$Q_G(x)$	A.1.4.6.1.3.1	Codebook gain quantizer function.
q_i	A.1.4.3.2.6	Coefficients of $Q_A(z)$.
$\hat{q}_i$	A.1.4.3.3.5	Coefficients of $\hat{Q}_A(z)$.
q'_i	A.1.4.3.2.6	Coefficients in $Q'(w)$.
$Q'(w)$	A.1.4.3.2.6	Function used in computing LSP frequencies.
$Q_{ti}(x)$	A.1.4.3.2.7	Quantizer without limiting for the ith LSP frequency.
$Q_{wi}(x)$	A.1.4.3.2.7	Quantizer for the ith LSP frequency.
Q_{wi}^{max}	A.1.4.3.2.7	Maximum LSP quantization level for the ith coefficient.
$R_i(0)$	A.1.4.4.1	First autocorrelation coefficient for the ith frame.
$R(k)$	A.1.4.3.2.3	kth autocorrelation coefficient.
SD	A.1.4.6.1.3.2	Random number used to generate CBSEED in a Rate 1/8 packet.
$s(n)$	A.1.4.3.2.2	Input speech samples corresponding to the frame or subframe with DC removed.
$s_d(n)$	A.1.4.8	Speech reconstructed by the receiving vocoder.
$s_w(n)$	A.1.4.3.2.2	Windowed input speech signal.
$SCALE_{fin}$	A.1.4.8.5	Final scale factor for the adaptive postfilter in the receiving vocoder.
$SCALE_{init}$	A.1.4.8.5	Initial scale factor for the adaptive postfilter in the receiving vocoder.
SM	A.1.4.3.3.3	Low-pass filter coefficient for the LSP frequency low-pass filter.
$T_k(B_i)$	A.1.4.4.1	Adaptive rate thresholds.

Table A.1.4.13-1. Summary of Service Option 1 Notation (Part 5 of 5)

Parameter	Section	Name/Description
w_i	A.1.4.3.2.6	LSP frequencies.
$\tilde{w}_i$	A.1.4.3.3.1	Regenerated LSP frequencies.
$\hat{w}_i$	A.1.4.3.3.3	$\tilde{w}_i$ after stabilization and filtering.
$\hat{w}'_i$	A.1.4.3.3.4	$\hat{w}_i$ after interpolation.
$\Delta\tilde{w}_{min}$	A.1.4.3.3.2	Minimum LSP frequency spacing.
$W_H(n)$	A.1.4.3.2.2	Hamming window.
$W(z)$	A.1.4.5.1	Perceptual weighting filter.
$x(n)$	A.1.4.5.1.1	$e(n)$ filtered by $W(z)$.
$y_d(n)$	A.1.4.8.3	Formant filter output.
$y_I(n)$	A.1.4.6.1.1	$c_I(n)$ filtered by $H(z)$.
$y_L(n)$	A.1.4.5.1.1	$p_L(n)$ filtered by $H(z)$, assuming $H(z)$ has zero initial state.
z	All sections	z transform variable.
ζ	A.1.4.5.1	Perceptual weighting parameter used in $W(z)$.

No text.