Chapter 2
Ultra Low Bit-Rate Coders

In this chapter, we present the definition and principles of ultra-low bit-rate coders. Here the emphasis is to point to the fact that this class of coders is typically the 'vocoders', which are 'parametric' coders that are essentially linear-prediction (LP) based vocoders. This is in contrast to the 'waveform' coders, which operate at the higher bit-rates. Among the various frameworks employed for realizing ultra low bit-rate speech coding, we restrict our attention to the generic linear-prediction (LP) based vocoding, (of which the LPC-10 coder [T82] sets the baseline) mainly for its rather ubiquitous adoption to a range of segment quantization vocoders as well as for its simplicity and effectiveness in establishing the low bit-rate operation and quality of coded speech. Here, the parameterization is based on the linear prediction filter model of human vocal tract wherein the linear prediction parameters (such as reflection coefficients, log-area-rations, line-spectral-frequencies, etc.) characterize the spectral shape of every frame of speech (typically of 20 ms duration) and the prediction error (residual) characterizes the excitation source of the speech production model. Coders have evolved to efficiently model and quantize both these information, i.e., the linear prediction parameters and the residual of each frame of the speech signal. However, by and large, the ultra low bit-rate coders have focused mainly on how to efficiently quantize the linear-prediction parameters either one frame at a time or by taking them as a sequence of frames together in the form of a 'segment'.

The main part of the book will focus on the various quantization schemes that have evolved towards quantizing the spectral parameters at the target bit-rates (of ultra low bit rate speech coding at less than few hundreds of bits/second). This chapter is organized to cover the various techniques in segment quantization, both conceptually and chronologically, as can be summarized under the following broad conceptual categories.

1. Vector quantization (VQ) and Matrix quantization (MQ) e.g., the VQ-LPC and MQ-LPC systems at 300–800 bps, viewed as fixed length segment vocoders.
2. Structure of generic variable length segment vocoders comprising various components and steps, such as feature extraction (e.g. LP analysis), automatic

© The Author 2015

V. Ramasubramanian, H. Doddala, *Ultra Low Bit-Rate Speech Coding*,
SpringerBriefs in Speech Technology, DOI 10.1007/978-1-4939-1341-1_2

segmentation, segment quantization (or joint segmentation and quantization), segment codebooks, duration modification, residual (or prosody) modeling, parameterization and quantization and synthesis. We provide a brief review of the various salient techniques proposed till date for each of these components and steps. Some of the techniques outlined include spectral transition measures, maximum likelihood segmentation, temporal decomposition for automatic segmentation, variable length segment quantization (VLSQ) using 2-level dynamic programming (DP) algorithm, joint segmentation and clustering algorithm (or the segmental K-means algorithm) for template segment codebook design, phoneme recognition and phonetic vocoders, template and HMM codebooks, space sampling for segment distortion and duration modification, pitch profile quantization, etc.

3. Rate-distortion constrained variable length segmentation and quantization, also referred to as R/D optimal linear prediction.
4. HMM based recognition-synthesis paradigms (sub-300 bps to 1 Kbps) focusing on HMM based phone class modeling and phone recognition in the HMM framework, followed by HTS based synthesis at the decoder.
5. ALISP units and refinements, focusing on the unit definition and modeling, segmentation and synthesis, with variants including definition of long synthesis units and short synthesis units by dynamic unit selection in a corpus based approach.
6. Unit selection paradigms, that mark a major departure from the notion of clustered codebooks or HMM codebooks to use of long continuous codebooks, in the form of single-frame vector codebooks or variable length segmental codebook, and which are used for segment quantization by unit-selection principles, as derived from TTS techniques.

On a related note, with regard to the LP parameter quantization, it is to be noted that we shall not attempt to review another important, and what can possibly be considered as mainstream class of quantization techniques, all set in the vector quantization framework (of quantizing vectors) and that operate at significantly much higher bit-rates, such as 24 bits/frame and with an effective bit-rate for 'spectral quantization' of 1,200 bps (using a frame-rate of 50 frames/s) and above. These are geared towards achieving high speech quality, referred to as 'transparent quality', characterized by the underlying spectral distortion of 1 dB [PK95]. In contrast, the focus here is exclusively on 'segment quantization', typically set in a segment vocoder framework, and operating at the lowest end of the bit-rate range of few hundreds of bits/second and less. In this regard, in addition to such high-rate vector quantizers (details of which can be found in the very comprehensive treatment of LP parameter quantization [PK95]), we also exclude from our consideration, another class of quantizers, also operating in the same bit-rate ranges, but based on exploiting inter-frame correlation between LP parameter vectors (LSF) such as [D04], [G07], [DP10].

2.1 Vector and Matrix Quantization

Figure 2.1 shows the basic LPC segment vocoder framework. Here, the speech signal is first analyzed at typical frame size 20 ms (in [T82], the standard LPC-10 vocoder uses a 22.5 ms frame size) yielding a LP parametric vector of dimension $p = 10$ corresponding to a LP order 10 in the LP analysis. Each frame also yields a residual associated with the LP parametric vector, which is further processed to derive the voiced/unvoiced decision, pitch and gain, which are the excitation parameters used to resynthesize speech at the decoder. LPC-10 quantizes the parametric vector (reflection coefficients) using non-uniform scalar quantization for each of the coefficients as described in [T82] with 41 bits/vector (and as shown in Fig. 2.2) and the excitation parameters (voicing, pitch, gain, sync) by scalar quantization with 13 bits/vector, yielding a total bit-rate of 54 bits/frame or 2.4 Kbps for a frame-rate of 44.4 frames/s. The quality of the LPC-10 coder is typically expressed as 'synthetic' quality, given in terms of subjective quality MOS score of 2.3.

As noted in Sect. 1.2, LPC-10 sets the basis for a class of low and ultra low bit-rate speech coding, all typically in the same LP vocoder framework, but essentially differing in the way the LP parameters are quantized using various techniques such as vector quantization, matrix quantization, variable length segment quantization, etc., progressively striving to reduce the effective bit-rate down from 2.4 Kbps, but keeping the LPC-10 quality as the target to achieve.

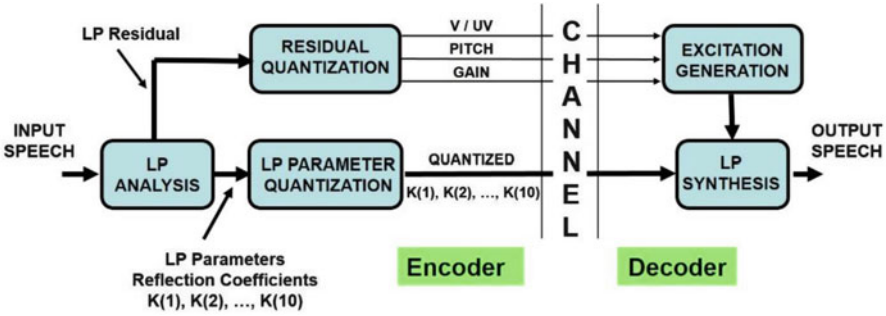


Fig. 2.1 Generic structure of the LPC-10 vocoder

	Pitch/ Voicing	Gain	Sync	K(1)	K(2)	K(3)	K(4)	K(5)	K(6)	K(7)	K(8)	K(9)	K(10)	Total	Error Correction
VOICED	7	5	1	5	5	5	5	4	4	4	4	3	2	54	0
UNVOICED	7	5	1	5	5	5	5							33	20

Fig. 2.2 Bit-rate allocation for quantization of reflection coefficients, pitch and gain in LPC-10 vocoder

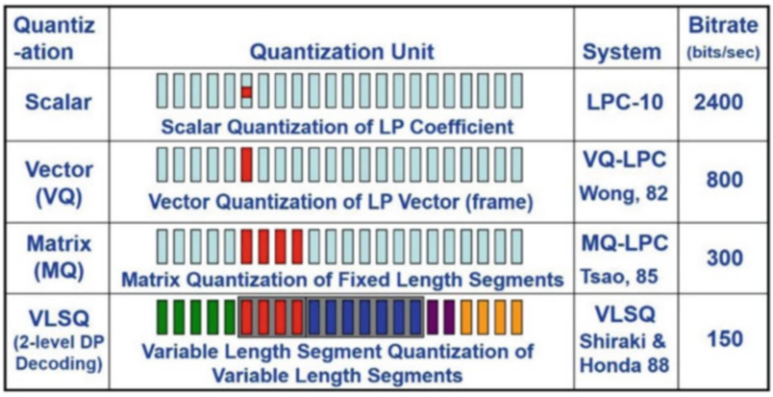


Fig. 2.3 Schematic of the various quantization schemes for LP parameter quantization and corresponding bit-rates

Figure 2.3 shows a schematic of the manner in which the LP parameter vector is quantized by the various quantization techniques referred above, namely, scalar quantization, vector quantization, matrix quantization and variable length segment quantization. Figure 2.4 shows a schematic of the feature space of the LP parameter vectors, where the input LP parameter vector sequence is quantized differently by a vector quantizer, matrix quantizer and variable length segment quantizer. In the following, we note these briefly, with reference to Figs. 2.3 and 2.4.

It can be seen that scalar quantization (as in the basic LPC-10 vocoder [T82]) quantizes each co-efficient separately, while vector quantization quantizes the entire LP parameter vector. This is shown in row 2 in Figs. 2.3 and 2.4a. The principles of VQ that lead to rate-distortion advantage over scalar quantization are well established, such as based on exploiting the higher dimensionality of vectors, linear and non-linear dependencies between the vector components, and vector pdf shape [M85]. The VQ-LPC coder [W82, M85, WJC83] marked an important milestone in ultra low bit-rate coding by applying the then emerging concept of vector quantization (VQ) to quantize the LP parameters as a vector for each frame of speech as against the conventional scalar quantization of the parameters as in the standard LPC-10 vocoder [T82]. This brought about a remarkable reduction of the bit-rate from 2.4 Kbps to 800 bps while preserving the quality of the LPC-10 vocoder.

A natural extension of quantizing a vector to a vector-of-vectors (or also referred to as 'fixed length segments'), saw the emergence of matrix-quantization based LPC-10 system, which reduced the bit-rate to 300 bps while preserving the quality of coded speech as same as that of LPC-10 [WJC83, TG85]. This is shown as row 3 in Figs. 2.3 and 2.4b. Other variants of the basic matrix quantization concept have also been proposed, notably, the multi-frame coding schemes [KCT91, KCT92] at 600–800 bps, [ML92] at 800 bps, [MLG95] at 800 bps and the matrix product quantization [Br95] at 300 to 700 bps. Such multi-frame coding paradigms

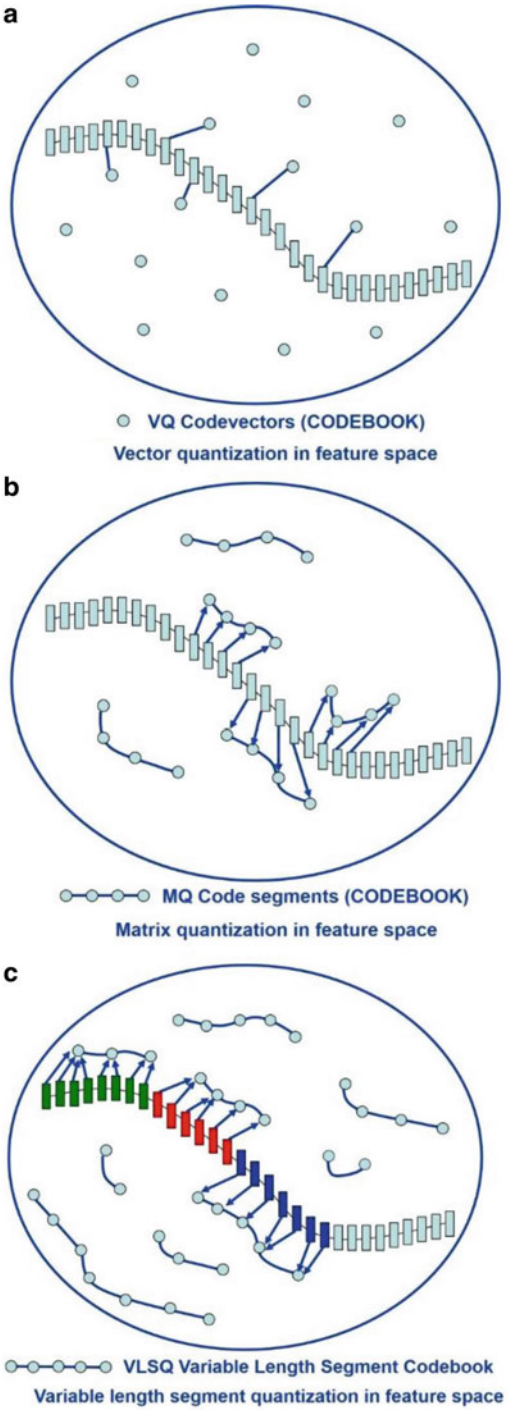


Fig. 2.4 Vector, matrix and variable length segment quantization in feature space

approximate a matrix of LP parameters (also called a super-frame) by a few frames as anchor points, with the other frames being differentially encoded or interpolated at the receiver. Among these, [Br95] employed a more rigorous formulation of representing the reconstruction matrix Y (of an input matrix X of m successive LPC parameter vectors to be quantized) as a product of a diagonal centroid matrix S representing the average parameter vector of the matrix and a temporal contour represented as a contour matrix V, and arriving at an iterative framework to design the joint S and V matrix codebooks. This work also extended this formulation to variable size matrices, i.e., segments, as is the topic of discussion in the next section. A further generalization of the multi-frame encoder of [KCT91] is the combined quantization-interpolation (CQI) method of [LF93] where the 'target' vectors defining the break-points or the end-frames of variable length segments marked by spectral discontinuity are determined sequentially in a two-pass framework to realize spectral distortions lower than that of [KCT91] at bit-rates below 350 bps. As we shall note later, the maximum-likelihood formulation of [SS87] in Sect. 2.2.1.2 provides an optimal framework for such a scheme, as was explored and compared in [S94].

The next important development was the progression from quantizing 'fixed length segment' to 'variable length segments', as in [SH88]. These techniques exploited the variable durations of speech units (typically, phones) and then quantized them efficiently using structured or unstructured 'segment' codebooks. Segment vocoders based on variable-length segment quantization has provided the means of achieving low to ultra low bit-rates in the range of 800 to 150 bps while offering intelligible speech quality [RSM83, RWR87, SH88, HS92]. This is illustrated in row 4 in Figs. 2.3 and 2.4c. We shall consider this class of quantization in a more general setting that is referred to as 'segment vocoders', in the following section in a little more detail, and provide an overview of a wide variety of ultra low bit-rate quantizers in this framework.

2.2 Segment Vocoders

The basic functioning of a segment vocoder framework is given in Fig. 2.5 and can be described as made of the following main components and/or steps:

1. **Automatic segmentation**: Segmentation of input speech (a sequence of LP parameter vectors) into a sequence of variable length segments (also referred to as 'units').
2. **Variable-length segment quantization**: Segment quantization of each of these segments using a segment codebook $C = (c_1, c_2, \ldots, c_N)$ and transmission of the best-match code—segment index and input segment duration.
3. **Joint segmentation quantization**: While the early systems used automatic segmentation and segment quantization as separate steps, (i.e., segmentation yielded the variable length segments that are further quantized), most segment

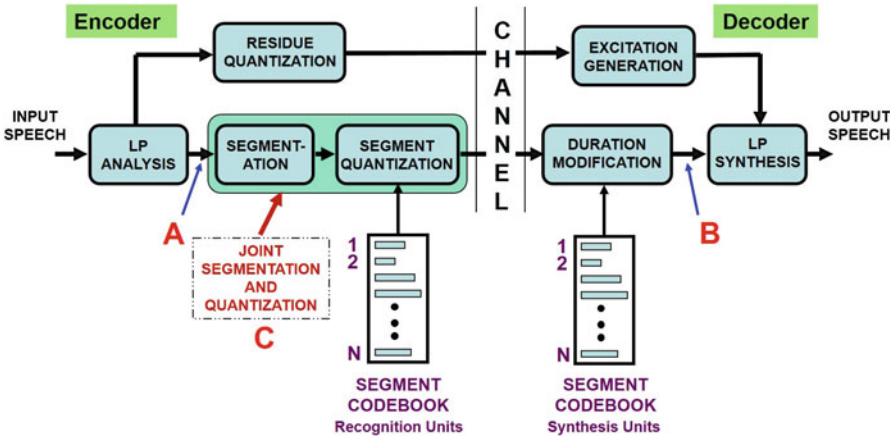


Fig. 2.5 Generic structure of a segment vocoder using LP analysis (at the encoder) and LP synthesis (at the decoder)

vocoders that evolved subsequently employed techniques that can be referred to as 'joint segmentation quantization', wherein the segmentation and quantization are combined within a single step for a given segment codebook, essentially solving what can be called a segmentation and labeling using methods derived from speech recognition. In Fig. 2.5, this is shown as block C that combines Step 1 and 2 above.

4. **Segment codebook:** The segment codebook of size N is typically made of N variable length segments or statistical models of segmental units such as HMMs, that are derived from training speech (long sequence of LP vectors) using special techniques such as joint segmentation and clustering procedures (or segmental K-means algorithm) and acoustic modeling techniques in speech recognition that yield the HMMs. Note that in Fig. 2.5, we have referred to the segment codebook at the encoder as 'recognition units' and the one at the decoder as the 'synthesis units'. As we will see in subsequent sections detailing this aspect, these are identical codebooks in most segment vocoders, particularly those using templates [RSM82b, RSM83, RWR87, RS04] and in the HMM based recognition/synthesis systems [T98, H03, MTK98]. However, these could also be different codebooks, differing in terms of the representation of the segments in the codebook (as long as each segment is associated with the same underlying acoustic properties); an example is [MBCC01, MGC01], where the recognition units are HMMs derived from LPCC representation, while the decoder has synthesis units that allow either LP synthesis or HNM synthesis. Other examples include [C08, CMC08a, CMC08b, PSVH10] (using syllable HMMs for segment quantization at the encoder and template based synthesis at the decoder) [LC01, LC02] (MFCC units for unit-selection based segment quantization at the encoder and HNM based synthesis at the decoder).

5. **Duration modification:** The code-segment corresponding to each received code-segment index is time normalized (duration modified) to match the duration of the corresponding input segment.
6. **Residual modeling and quantization:** The residual obtained by LP analysis is also parameterized, quantized and transmitted; the residual decoder reconstructs the residual to be used for synthesis in Step 6.
7. **Synthesis:** Synthesis of speech by LP synthesis using the code-segment time-normalized to match input segment duration.

The various segment vocoders proposed till date differ primarily in some or all of these aspects. In the following, we briefly describe each of the above components in more detail, primarily providing a review of how various segment quantization techniques for ultra low bit-rate coding differ with reference to these specific components and steps outlined above.

2.2.1 Automatic Segmentation

Here, we briefly review the automatic segmentation techniques in various segment vocoders that derive a variable length segmentation prior to segment quantization.

Much of the basic architecture in the segment vocoder framework was laid by the work at BBN [MCRK77, SKMKZ79, SKMS80, RSM82a, RSM82b, RSM83, RW85, RWR87, M06]. Some of these early work followed a segment-then-quantize approach, for example in [RSM82b, RSM83] where automatic segmentation was done using a simplistic spectral transition measure to generate diphone-like units which are used in both segment codebook design and segment quantization. In contrast, [RWR87] employed a combinatorial search for the best segmentation of a given block of input speech, under the constraint that segment durations are to be from a finite set, and optimized to minimize the quantization error for the block.

An interesting work by Svendsen [S94] showed that using an VQ codebook for quantizing segments derived automatically by the maximum-likelihood (ML) segmentation [SS87] yields a bit-rate reduction by a factor of 2, while preserving the speech quality. This clearly established the essence of segment quantization, namely, that it is more efficient to quantize a segment as whole, if the segment corresponds to an acoustic unit such as a phone (which comprises of a steady-state and therefore can be quantized parsimoniously by a single vector or a code-segment from a segment codebook).

[RS04] extended these early studies above and explored automatically derived phone and diphone units using spectral-transition measures and phone-like units using maximum-likelihood segmentation [SS87] and showed that automatically derived phone-like units are more efficient than diphone-like units for typical segment codebook sizes (made of randomly selected phone-like or diphone-like units in the form of segments).

2.2.1.1 Spectral Transition Measure

The 'spectral transition measure' (STM) is based on the principle of measuring the spectral derivative at every frame instant. STM was adopted in early segment vocoders for diphone-like segmentation [RSM82b, RSM83]. We consider two types of STM as used in [RSM82b], namely, the d_1 and d_3 measures. These are defined as follows: Let $\boldsymbol{o}_n$ be the LP parameter vector at the nth frame. The STM at frame n, $d_i(n)$, is given by $d_i(n) = \|\boldsymbol{o}_n - \boldsymbol{o}_{n-i}\|^2, i = 1, 3$.

Figure 2.6 illustrates the STM profile with respect to phone-like and diphone-like segments and the corresponding peak/valley picking as described below. $d_1(n)$ as a function of n exhibits peaks at fast spectral transitions (such as from one phone to another) and valleys at steady state regions (such as within a vocalic segment). $d_3(n)$ gives a smoother measure of the spectral derivative. Thus, peak-picking of $d_1(n)$ or $d_3(n)$ locates transitions or phone boundaries and results in a phone-like segmentation. Picking the minima (valleys) of these functions locates a frame within steady-state regions that has maximum local stationarity and corresponds to a diphone boundary. Successive peaks therefore mark phone-like (PL) segments and successive valleys mark diphone-like (DPL) segments.

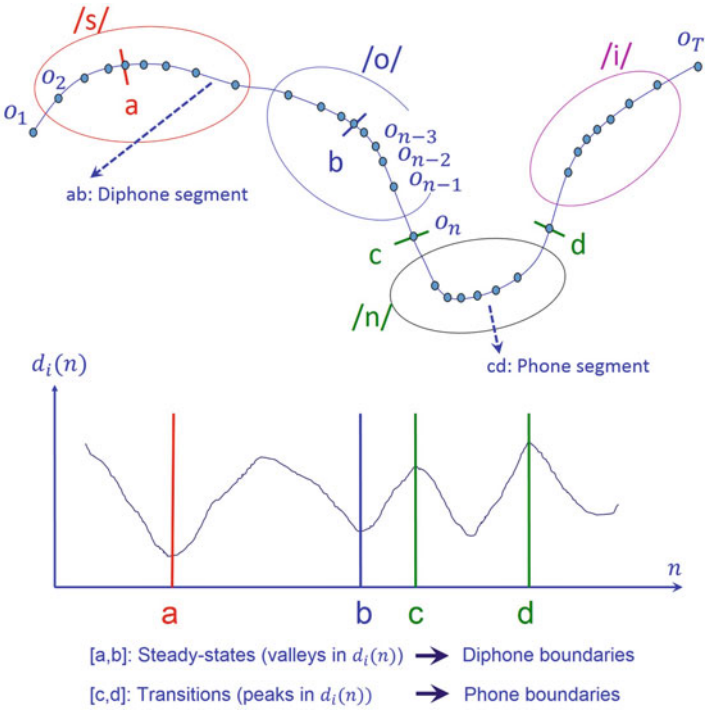


Fig. 2.6 Spectral transition measure (STM) based definition of phone-like and diphone-like segmentation and corresponding STM profile used for peak/valley picking

[RS04] used an extrema picking algorithm (EPA) for peak- and valley-picking on $d_1(n)$ and $d_3(n)$ functions. This algorithm employs a threshold (δ) to detect the extrema (peaks and valleys) alternatingly in a left-to-right scanning. The algorithm can be stated as follows:

Search
for a peak (valley) by repeated updating of current maximum p (current minimum v) every time a local maximum (minimum) is detected
Until
a function value smaller than $(1-\delta)p$ *(larger than* $(1+\delta)v$*) is encountered.*
After this,
Go to Search *and start searching for a valley (peak).*

While small values of δ (close to 0) result in over-segmentation, large values of δ (close to 1) result in under-segmentation and needs to be optimized for a desired segment rate. Thus, the threshold δ used in the extrema-picking-algorithm plays a crucial role in determining the quality of segmentation and hence needs to be optimized to yield good segmentation match as defined above. For this purpose, [RS04] used the segment rate (number of segment/second) as the primary measure to be matched. For instance, TIMIT has a phone-rate of $R = 12.5$ phones/s, as measured over 300 sentences. In STM, [RS04] set δ to a value that results in this segment rate. The optimal δ corresponding to a segment rate of 12.5 segments/s also results (automatically and interestingly) in the highest percent match (between the automatically determined segment boundaries and the true segment boundaries within a specific tolerance limit) as well as the lowest percent insertion and deletion values. Based on segmentation applied on TIMIT sentences, it was shown that d_1 is well suited for detecting fast spectral transitions such as phone boundaries and hence in better phone-like segmentation than d_3. Conversely, the smoother d_3 is more suited for a diphone-like segmentation than d_1, where it is necessary to detect steady-states by valley-picking. As a consequence, [RS04] used STM (d_1) for phone-like (PL) segmentation and STM (d_3) for diphone-like (DPL) segmentation in the overall vocoder.

2.2.1.2 Maximum-Likelihood Segmentation

Let a speech utterance be given by $O_1^T = (\boldsymbol{o}_1, \boldsymbol{o}_2, \ldots, \boldsymbol{o}_T)$, which is a LP parameter vector sequence of T speech frames, where, $\boldsymbol{o}_n$ is a p-dimensional parameter vector at frame 'n'. The segmentation problem is to find 'K' consecutive segments in the observation sequence O_1^T. Let the segment boundaries be denoted by the sequence of integers $B = (b_0, b_1, \ldots, b_{k-1}, b_k, \ldots, b_K)$. The kth segment starts at frame $b_{k-1}+1$ and ends at frame b_k; $b_0 = 0$ and $b_K = T$. This is illustrated in Fig. 2.7.

The maximum likelihood (ML) segmentation is based on using the piecewise (quasi) stationarity of speech as the acoustic criterion for determining segments. The criteria is to obtain segments which exhibit maximum acoustic homogeneity within

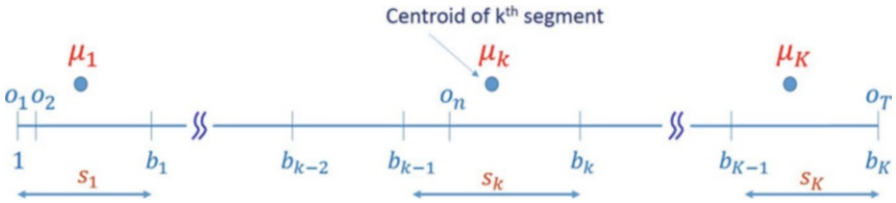


Fig. 2.7 Schematic of maximum-likelihood segmentation

their boundaries. The acoustic inhomogeneity of a segment is measured in terms of an 'intra-segmental distortion', given by the sum of distances from the frames that span the segment, to the centroid of the frames comprising the segment. For a given K, the optimal segmentation $B^* = (b_0^*, b_1^*, \ldots, b_K^*)$ is obtained so as to minimize the sum of intra-segment distortion over all possible segment boundaries, i.e., minimize

$$D(K,T) = \sum_{k=1}^{K} \sum_{n=b_{k-1}+1}^{b_k} d(\boldsymbol{o}_n, \mu_k)$$

which can be given as,

$$B^* = \arg\min_{B} D(K,T)$$

where, $D(K,T)$ is the total distortion of a K-segment segmentation of $O_1^T = (\boldsymbol{o}_1, \boldsymbol{o}_2, \ldots, \boldsymbol{o}_T)$; μ_k is the centroid of the kth segment consisting of the spectral sequence $O_{b_{k-1}+1}^{b_k} = \{\boldsymbol{o}_{b_{k-1}+1}, \ldots, \boldsymbol{o}_{b_k}\}$ for a specific distance measure $d(\cdot,\cdot)$. For the Euclidean distance 'd', μ_k is the average of the frames in the segment $O_{b_{k-1}+1}^{b_k}$. This is illustrated in Fig. 2.7.

The optimal segment boundaries are solved efficiently using a dynamic programming (DP) procedure [SS87, SRS02] using the recursion

$$D(k, b_k) = \min_{b_{k-1}} [D(k-1, b_{k-1}) + \Delta(b_{k-1}+1, b_k)]$$

for all possible b_{k-1}; $D(k, b_k)$ is the minimum accumulated distortion up to the kth segment (which ends in frame b_k), i.e., $D(k, b_k)$ is the minimum distortion of a segmentation of $(\boldsymbol{o}_1, \boldsymbol{o}_2, \ldots, \boldsymbol{o}_{b_k})$ into k segments; $\Delta(b_{k-1}+1, b_k)$ is the intra-segment distortion of the kth segment $O_{b_{k-1}+1}^{b_k}$. This is illustrated in Fig. 2.8. The segmentation problem is solved by invoking (2) for $D(K,T)$; this is computed efficiently by a trellis realization. The optimal segment boundaries $(b_0^*, b_1^*, \ldots, b_K^*)$ are retrieved by backtracking on the trellis along the optimal alignment path corresponding to $\min\{D(K,T)\}$.

Figure 2.9 shows a schematic of the feature space where the input feature sequence corresponding to the word 'emotion' is segmented by ML-segmentation into corresponding phone-like units.

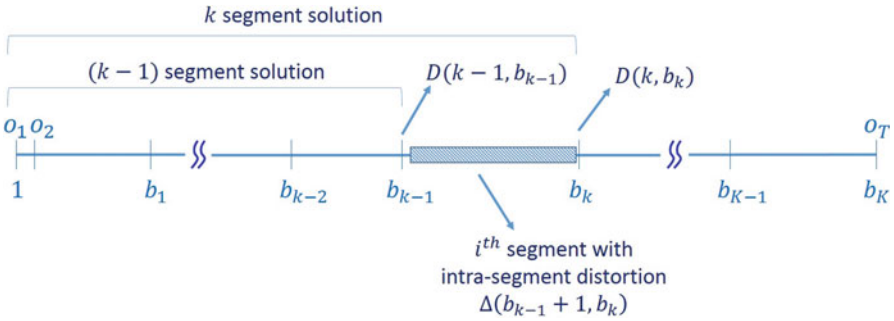


Fig. 2.8 Recursive structure employed in maximum-likelihood segmentation

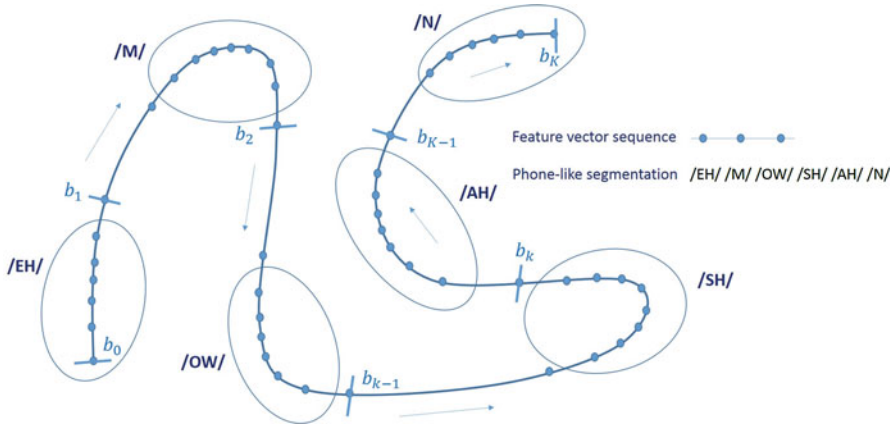


Fig. 2.9 Illustration of a typical phone-like segmentation derived by ML segmentation in a feature space

2.2.1.3 ML Segmentation: Duration Constrained (ML(DC))

By definition, ML segmentation produces a segmentation where each segment is maximally homogenous; when the segment rate equals the phone-rate of natural speech, the resulting segments will be quasi-stationary and would correspond to the steady-state regions of phonetic units. However, even for a correct segment rate, ML segmentation can result in segment lengths which are unnaturally short (1 frame long) or long (up to even 70 frames). Such segments will be distorted significantly during segment quantization and result in poor vocoder performance.

In the distribution of phone durations in TIMIT database, nearly 95 % of the labeled phonetic segments are in the range of 1-20 frames. In order to limit the segment lengths of ML segmentation to such a meaningful range (of actual phones durations), [RS04] modified the ML segmentation to have 'duration constraints'. Here, the optimal segments are forced to be within a duration range of $[\alpha, \beta]$, where

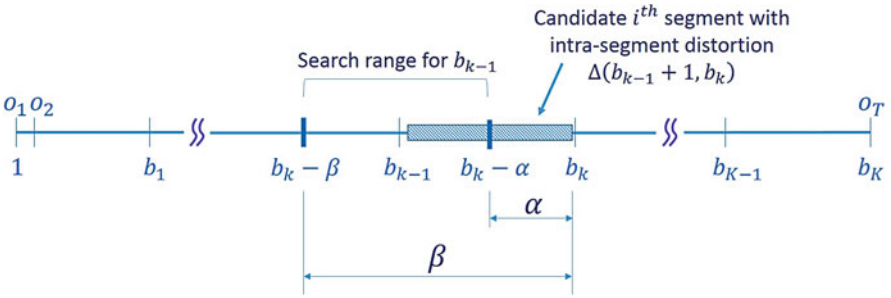


Fig. 2.10 Recursive structure employed for duration-constrained ML segmentation

Fig. 2.11 Different segmentation techniques in STM and ML framework and corresponding phone-like and diphone-like units explored in [RS04]

	Segmentations			
	STM		ML	
Units	d_1	d_3	UC	DC
PLU	○	○	○	○
DPLU	○	○		

UC: Unconstrained
DC: Duration-constrained

α and β are the minimum and maximum lengths possible (in frames). Segments of lengths $<\alpha$ and $>\beta$ are not generated at all. This is achieved by restricting the candidate boundaries in the search for optimal segment boundaries in (2) as follows, and as illustrated in Fig. 2.10:

$$D(k, b_k) = \min_{b_k-\beta \le b_{k-1} \le b_k-\alpha} [D(k-1, b_{k-1}) + \Delta(b_{k-1}+1, b_k)]$$

This also has the advantage of reducing the computational complexity of ML segmentation from $O(T^2)$ to $O(lT)$, where $l=\beta-\alpha+1$ with typical values of $[\alpha, \beta] = [2, 20]$.

This work [RS04] evaluated the STM and ML segmentations for phone-like units (PLU) and diphone-like units (DPLU) as shown in Fig. 2.11. It used various segmentation accuracy measures and segmental SNR to compare the performance of the different phone-like and diphone-like units (by retaining the residual without modeling or quantization), and showed that ML segmentations realize phone-like units which are significantly better than those obtained by STM in terms of match accuracy with TIMIT phone segmentation as well as actual vocoder performance measured in terms of segmental SNR. Further, it was shown that the phone-like units of ML segmentations also outperform the diphone-like units obtained using STM in early vocoders. The resulting segment vocoder had an average bit-rate of 300 bps, with very high intelligibility when used in a single-speaker mode. The interesting observation to be made in this context is that phone-like units are typically characterized by segments between two transitions with a stable steady-

state at its center. Hence these units have a good means of representing a phonetic unit for a given codebook size, despite the high invariability arising in the transitional parts arising from context dependency. In contrast, the diphone units, defined from the steady-state of one phoneme to the steady-state of another, has low variability and hence represent a particular diphone better than a phone-like unit would. However, considering that the number of diphones is large $O(M^2)$ for M phones, the codebook size needs to be larger than that of phone-like units to ensure adequate coverage and representation of all possible diphones. Thus, for a given codebook size of the order of 8,192, as considered in this work, the phone-like units have the possibility of being represented by several segments belonging to a given phone-class (e.g. 128 segments = 8,192/64, for number of phones as 64) thereby accounting for contextual variability to a certain extent; on the other hand, this codebook size is still inadequate for representing diphone-like units (e.g. $64^2 = 4{,}096$), which approximately allows only two segments of a diphone class to be present in the codebook. (Of course, these numbers are approximate to allow reasoning with the scenarios, and the units are bound to be distributed non-uniformly). This gives a reasonable explanation for why the above work reported a better performance of the phone-like units for the kind of codebook sizes employed.

2.2.1.4 ML Segmentation: A Generalized Basis

An interesting observation to note here is that the ML segmentation formulation outlined above sets the basis for at least five different kinds of segmentation and quantization techniques that are discussed in this chapter, namely,

1. Distortion constrained segmentation and vector quantization of [S94] pointed to in Sect. 2.2.1 (also generalizing over the sequentially determined combined quantization-interpolation method of [LF93] indicated in Sect. 2.1) and discussed in some detail under the R/D optimal techniques in Sect. 2.3.4.
2. Joint segmentation and quantization technique forming the core of the variable length segment quantization [SH88], and derived based on the 2-level dynamic programming algorithm discussed in Sects. 2.2.3, 2.2.3.2 and 2.2.3.3.
3. R/D optimal algorithm of [PV00] discussed in Sect. 2.3.1.
4. Rate-distortion constrained segmentation in the variable-to-variable length vector quantization (VVVQ) formulation of [CL94] discussed in Sect. 2.3.2.
5. Multigram formulation (and sequence segmentation) of [B95] and reinterpretation with reference to VVVQ in [BCC97], discussed in Sect. 2.3.3.

2.2.1.5 Syllable-like units and other segmentations

In an interesting departure from classical units such as phone-like and diphone-like units, [C08, CMC08a, CMC08b, PSVH10] proposed the use of syllable as the unit of segmentation and quantization and derive syllable-like units using group-delay based automatic segmentation [NM04, MY11] in a syllable-based segment vocoder

framework. Other work to use syllable based unit definition for segmentation are [CKBC99] which detects local maxima of the function of 'sonority decrease' to further identify and select syllables, (this work uses syllables as alternative to the ALISP units, discussed separately in Sect. 2.5), and [HN89] which uses syllable as an unit for recognition and synthesis, and defines reference patterns of syllable units using hand labeling.

In a study of segmentation techniques for segment vocoders, [C05] considers the segmentation algorithm by [v91, BS00], reporting spectral distortions of 2.5–1.5 dB at bit-rates in the range of 100–400 bps. [v91] define a segmentation by treating phonemes as stationary regions and detecting the transitional boundaries of a phoneme marked by a drop in the normalized correlation between frames below a specified threshold. [BS00] use a vowel-spotting approach by peak-detection of average magnitude, by viewing vowels as characterized by larger average magnitudes than nearby frames.

2.2.1.6 Temporal Decomposition

Temporal decomposition is an important technique [A83] to model the articulatory dynamics of speech production, where speech is described as a result of a sequence of distinct articulatory gestures towards and away from articulatory targets, the influence of the neighboring gestures thereby contributing to the co-articulatory effects commonly underlying all continuous speech. Atal [A83] proposed temporal decomposition as a means of describing a sequence of parametric vectors representing continuous speech as a linear combination of the underlying targets, by solving for a set of target vectors which when linearly combined by (a small number of) overlapping target functions, can approximate a parametric vector at each frame. Among the further work in this direction [V89, VM89, BA91], Van Dijk-Kappers [V89] studied the phonetic relevance of temporal decomposition. Temporal decomposition has evinced interest for application in speech recognition and speech coding, notably very low bit-rate speech coding, given its appealing property to approximate continuous speech in terms of a parsimonious set of representations in the form of the target vectors and target (overlapping) functions, even while being amenable to synthesis by the very nature of the formulation being to approximate each vector in the sequence by the linear combination of a limited number of target vectors. In this regard, we note here that the temporal decomposition framework has led to significant work in the context of very low bit-rate speech coding, such as the 450–600 bps vocoders of [CS90, CS91, CS93] yielding speech, that, when compared to the 2,400 bps LPC, is very intelligible and more natural sounding speech, the 600 bps range vocoder yielding intelligible speech in [GD96, GDB97b], comparative study of different spectral representations of [GDB97a] including LPC, reflection coefficients, LAR, cepstrum, and band filters for event detection with respect to their decomposition suitability (not for their phonetic relevance, but the degree of reconstruction accuracy possible), 1,000–1,200 bps high fidelity compression in [GDS98], the 996 bps vocoder of [Sung98] using temporal decomposition on LSF parameters

yielding reasonable speech quality compared to the 2,400 bps LPC10e and a related work of [P95] in terms of step decomposition of speech. An important application of temporal decomposition is in the definition and modeling of ALISP units, which has been successfully applied for very low bit-rate speech coding, and which has been discussed separately in Sect. 2.5.

In the context of deriving segments by an automatic segmentation algorithm, it is to be noted that the problem of automatic segmentation to yield various types of meaningful acoustic segments is well studied with numerous techniques proposed till date. While the above sections did touch on some of the salient segmentation techniques specifically in the context of ultra low bit-rate speech coding, a broader discussion into all other techniques is beyond the scope here. Details of a wide range of automatic segmentation techniques can be found in the very comprehensive review of Vidal and Marzal [VM90].

2.2.2 *Segment Quantization*

Segment quantization as an explicit step is applicable only when the automatic segmentation is done as a first step to result in variable-length segments, and hence is specific to only those techniques discussed above under automatic segmentation.

Once the variable length segments are obtained, each of these segments s_k, $k = 1, \ldots, K$ is quantized using the best segment c_{q_k} in the segment codebook $C = (c_1, c_2, \ldots, c_N)$ defined as the segment yielding the lowest segmental distortion. This is shown in Fig. 2.12.

Quantizing an input segment to the best matching segment from the segment codebook calls for the definition of a distance metric to measure the segmental distortion between two arbitrary variable-length segments (the input segment to be quantized and a segment from the segment codebook). Such a measure needs to account for non-linear temporal and acoustic variability between two such segments, and ideally, this is best computed by dynamic time warping (DTW) well defined and applied for speech recognition and widely used in the ALISP units based segment quantization discussed in detail in Sect. 2.5. Linear warping, as a computationally simpler approximation to a full-fledged DTW as well as the so-called space-sampling based matrix-distortions have also been applied.

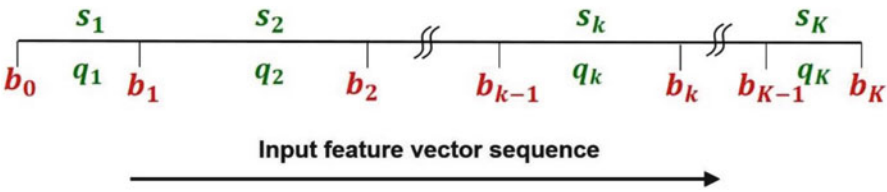


Fig. 2.12 Segmentation and quantization

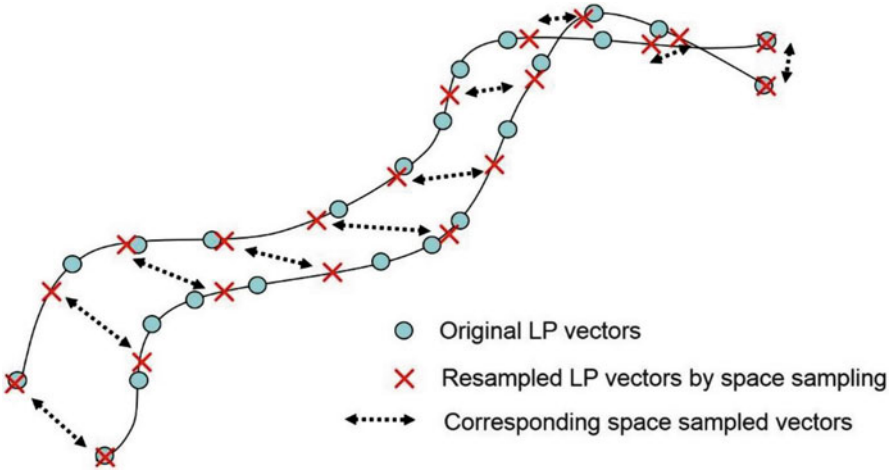


Fig. 2.13 Schematic of space sampling and correspondence between space-sampled vectors between two trajectories

For instance, in the primary work from BBN [RSM82b, RSM83, RWR87] which performed a diphone-like segmentation, they employed a space-sampling technique by which both the input segment and each of the variable-length segment in the segment codebook were re-sampled in the feature space (e.g. LAR vector space) to yield a fixed number of vectors that are equally spaced along the trajectory of the original segments. This can be visualized as shown in Fig. 2.13. The number of resampled vectors is set as 10, thereby converting each variable length segment to a fixed length segment (matrix) of length 10. Subsequently, a matrix distortion is computed as the sum of 10 Euclidean distortions, each between the corresponding LAR vectors of the resampled input segment and codebook segment. An input segment is then quantized to the codebook segment that has the minimum matrix distortion. While being computationally simple, this method circumvents a more exact and optimal dynamic time-warping kind of distance calculation, whose computation cost is proportional to the lengths of the input segment and each segment in the segment codebook being matched. The latter work of [RS04] also followed the same approach for segment quantization using a segment codebook for phone-like and diphone-like units.

In the case of the segment codebook being a HMM codebook, i.e., each entry is an HMM modeling a phone or phone-like or acoustic units, segment quantization takes the form of finding the best matching HMM from the HMM codebook, that has the maximum likelihood (usually, the Viterbi likelihood) for the input segment. While most of the HMM codebook based methods perform a transcription of the input speech in terms of the units that the HMMs model (and thereby falling under the joint segmentation and quantization approach, described in the next sub-section), the syllable based approach of [C08, CMC08a, CMC08b, PSVH10] used syllable HMM codebook, and performed a separate segment quantization

using this method of finding the best matching HMM for an input segment, which is a syllable-like segment derived from a first stage group-delay based segmentation, as pointed out in Sect. 2.2.1.

2.2.3 *Joint Segmentation Quantization*

We first give the problem definition of joint segmentation and quantization, before attempting a brief overview of different techniques that falls within this definition. In reviewing the class of segment vocoder that use joint segmentation and quantization as above, we also point to the nature of the segment codebook employed in the various joint segmentation and quantization work alongside here, considering the dependence of the joint segmentation and quantization (or segmentation and labeling) on the 'segment codebook'. We also devote a separate section for segment codebooks in order to highlight additional details as may be required in specific instances.

2.2.3.1 Basic framework

As the name implies, joint segmentation and quantization performs a segmentation and quantization of the associated segments simultaneously, i.e., given an input sequence of speech feature vectors (e.g. LPC parameter vectors), it obtains a set of segments each labeled by a segment from a segment codebook under some optimality consideration. This can be viewed as a 'segmentation and labeling' process, where each segment is labeled by a segment index from a segment codebook. Due to this, joint segmentation and quantization necessarily requires a segment codebook with which the segmentation and labeling is done.

Figure 2.12 shows a typical segmentation and labeling of a feature vector sequence $O = (\boldsymbol{o}_1, \boldsymbol{o}_2, \ldots, \boldsymbol{o}_T)$. Let the segment codebook used for such a segmentation and labeling be given by a collection of N variable length segments $C = (c_1, c_2, \ldots, c_n \ldots, c_N)$, where a segment c_n is a sequence of LP parameters of some length. Figure 2.12 shows the segmentation of the input feature vector sequence into K segments, defined by the segment boundaries $B = (b_0, b_1, \ldots, b_K)$. Each segment $s_k = (x_{b_{k-1}+1}, x_{b_k})$ is given a label q_k, where $q_k \in (1, N)$. Let the set of labels associated with the K segments $S = (s_1, s_2, \ldots, s_{k-1}, s_k, \ldots, s_K)$ be $Q = (q_1, q_2, \ldots, q_{k-1}, q_k, \ldots, q_K)$.

A typical segmentation is completely defined by

1. The number of segments K,
2. The segment boundaries for a given K, $B = (b_0, b_1, \ldots, b_{k-1}, b_k, \ldots, b_K)$ and,
3. The segment labels $Q = (q_1, q_2, \ldots, q_{k-1}, q_k, \ldots, q_K)$.

The problem of joint segmentation and quantization naturally needs to minimize the quantization error (or distortion) associated with each segment, and hence a measure of total segment distortion which, for a given (K, B, Q), can be expressed as

$$D = \sum_{k=1}^{K} d\left(s_k, c_{q_k}\right)$$

The optimal segmentation is determined as (K^*, B^*, Q^*) that minimizes D. Thus the solution for optimal segmentation can be given as

$$(K^*, B^*, Q^*) = arg \min_{K,B,Q} D = \arg \min_{K,B,Q} \sum_{k=1}^{K} d\left(s_k, c_{q_k}\right)$$

This is clearly a combinatorial search problem, which by brute force search will require first hypothesizing a K, and then hypothesizing a B with K segment boundaries, and for each resulting segment, obtaining the best matching codebook segment c_{q_k} and repeating this for all K and B, until the minimum overall distortion D is found. Fortunately, this problem is already well solved in the context of speech recognition, where the above definition of segmentation and labeling is exactly what corresponds to the now well established 'connected word recognition' (CWR) problem [RJ93], where we are given a input feature vector sequence of a test speech spoken in a continuous fashion made of a sequence of words drawn from a fixed vocabulary of N words and it is required to determine the sequence of words spoken, i.e., that best explains the feature vector sequence according to a distance measure or likelihood measure with respect to a set of N word models that the CWR system has [which could be template models or hidden Markov model (HMM)]. The CWR problem has at least three types of algorithmic solutions [RJ93], namely,

(a) 2-level dynamic programming (DP) algorithm
(b) Level-building algorithm
(c) One-pass DP algorithm

In light of the above basic definition of joint segmentation and quantization, it can be noted that 'segment quantization' techniques in segment vocoders essentially performed a 'connected segment recognition' by which they determined the optimal segment boundaries (and hence the segment lengths) and the segment labels that were transmitted and used for reconstructing speech at the decoder after duration normalization.

2.2.3.2 Shiraki and Honda Variable-Length Segment Quantization

At this point, it is appropriate to refer to the work of Shiraki and Honda [SH88, HS92] as who first established the notion of joint segmentation and quantization in the context of variable length segment quantization. While the joint segmentation

and quantization algorithm used by Shiraki and Honda [SH88] was essentially the 2-level DP algorithm proposed much earlier by Sakoe [S79] for connected word recognition, the importance of the variable length segment quantization work in [SH88] lies in the fact that [SH88] defined and addressed the design of 'optimal' variable length segment codebooks of any desired size, from continuous speech training data using an iterative 'joint segmentation and clustering' algorithm, which in speech recognition literature had been developed independently and referred to as the segmental K-means (or SKM) algorithm [RWJ86, RJ93]. We shall illustrate this codebook design algorithm in Sect. 2.2.4 as part of a discussion on 'segment codebook'. [SH88] also defined segment to segment distortions (within the joint segmentation and quantization framework) using linear interpolation and warping transform matrices.

2.2.3.3 2-Level DP Framework for Joint Segmentation and Quantization

We give now a brief note on the 2-level DP algorithm employed by Shiraki and Honda [SH88] for performing joint segmentation and quantization, particularly with respect to how it can be viewed and realized as a variant of the ML algorithm described in Sect. 2.2.1.2. It was seen that the ML algorithm solves for the optimal segmentation boundaries as B^* given by,

$$B^* = \arg\min_{B} D(K,T)$$

where $D(K,T)$ is the sum of the K intra-segmental distortions each being defined as in the second summand below,

$$D(K,T) = \sum_{k=1}^{K} \sum_{n=b_{k-1}+1}^{b_k} d(\boldsymbol{o}_n, \mu_k)$$

In the ML formulation, the intra-segment distortion for a segment $O_{b_{k-1}+1}^{b_k} = \{\boldsymbol{o}_{b_{k-1}+1}, \ldots, \boldsymbol{o}_{b_k}\}$ is defined with respect to the centroid μ_k of this segment. Instead, assume that the ML formulation is to be posed as the joint segmentation and quantization problem, as defined above, such that each resulting segment has the lowest possible quantization distortion and the segments (from the solution for the segment boundaries) are to be determined so as to minimize the total distortion. In this case, we first need to take into account a segment codebook from which the best segment is to be chosen to optimally quantize a given segment. The ML formulation can now be redefined so as to yield the desired joint segmentation and quantization with respect to this explicitly defined codebook by performing two changes in the basic ML equations:

(a) The intra-segment distortion $\sum_{n=b_{k-1}+1}^{b_k} d(\boldsymbol{o}_n, \mu_k)$ is replaced by the minimum quantization distortion of the segment $O_{b_{k-1}+1}^{b_k} = \{\boldsymbol{o}_{b_{k-1}+1}, \ldots, \boldsymbol{o}_{b_k}\}$ with respect to the segment codebook $C = (c_1, c_2, \ldots, c_n \ldots, c_N)$, i.e., by defining $D(K,T)$ as

$$D(K,T) = \sum_{k=1}^{K} d_k^*$$

where, d_k^* is the minimum quantization distortion of the segment $O_{b_{k-1}+1}^{b_k} = \{\boldsymbol{o}_{b_{k-1}+1}, \ldots, \boldsymbol{o}_{b_k}\}$ given by

$$d_k^* = \min_{c_n \in C} d\left(O_{b_{k-1}+1}^{b_k}, c_n\right)$$

where $d\left(O_{b_{k-1}+1}^{b_k}, c_n\right)$ is the segmental distortion between segment $O_{b_{k-1}+1}^{b_k}$ and a code segment c_n in the segment codebook C, realized by an appropriate time-warping (such as an optimal dynamic time-warping [RJ93]) or by the linear warping defined by [SH88].

(b) Note that in the ML formulation, the overall distortion $D(K,T)$ decreases monotonically with K, reaching a value of 0 when $K = T$, i.e., the input feature vector sequence of T vectors is segmented into T segments, yielding the trivial solution of each segment being a single vector. However, interestingly, in the case of 2-level DP based segmentation and quantization (or connected segment recognition), when $D(K,T)$ is obtained for a range of K of interest (from considerations of minimum to maximum search range for the desired number of segments, in the extreme case giving the limits of 1 to T), it is possible to obtain an optimal K^* for which $D(K^*,T)$ is a minimum, i.e., the segment to segment matching requirement dictates and constrains the segmentation solution to an optimal number of segments for which the overall quantization distortion is minimum, with less or more number of segments leading to duration mismatches between the segments in the segment codebook and the input segments being quantized. This leads to the second modification to the ML formulation, that K^* is obtained as

$$K^* = \arg\min_K D(K,T)$$

Thus the 2-level DP can be realized as a reformulation of the ML-segmentation algorithm with the above two modifications, defined with respect to an explicit codebook (in the place of the implicitly defined 'centroid' as the approximation of a segment) with the joint segmentation and quantization solved as (K^*, B^*, Q^*) as above (Note that $Q^* = q_1^*, q_2^*, \ldots q_k^*, \ldots, q_{K^*}^*$ is obtained incidentally from K^* and

B^* as solved above, where once segment s_k is solved as $s_k = \left\{ \boldsymbol{o}_{b^*_{k-1}+1}, \ldots, \boldsymbol{o}_{b^*_k} \right\}$, the quantization index q^*_k for this segment is simply given as

$$q^*_k = arg \min_{n=1,\ldots,N} d\left(O^{b^*_k}_{b^*_{k-1}+1}, c_n \right)$$

2.2.3.4 One-Pass DP Algorithm

We now briefly describe the one-pass DP algorithm listed as one of the three algorithms for solving the connected word recognition (or 'connected segment recognition' in the context of joint segmentation and quantization) for two reasons—firstly that it is also called the now well-known Viterbi-decoding when the word model is a HMM and we shall shortly discuss phone HMM based phoneme recognition which forms the joint segmentation and labeling for a class of segment quantizers, and secondly, we will use the one-pass DP algorithm in a modified form in the subsequent chapters to realize the unified and optimal algorithms for unit-selection based segment quantization and this short introduction to this algorithm serves as a precursor to these following chapters.

Here, we will mainly illustrate how the solution looks like for the one-pass DP algorithm for the case when the segment codebook is a set of templates of variable length segments, rather than go into the actual algorithmic realization of the one-pass DP algorithm (which can be found in [RJ93, N84]). The one-pass DP algorithm is applied on the sequence of feature vectors (in the x-axis) and the set of templates in the segment codebook (in the y-axis), and derives the solution in the form of an optimal decoding path, as shown in Fig. 2.14, which is retrieved by a backtracking procedure at the end of applying a path growing procedure (made of recursions) from left to right (i.e., from the starting frame in the input feature vector sequence to the last frame). This path establishes the mapping between the input feature vector sequence and the templates, yielding the segmentation and labeling solution (K^*, B^*, Q^*) as follows: The discontinuities in the path represent a transition from one word to another, and the number of distinct such sub-paths yields the optimal number of segments K^*. The segment to segment transition, marked by the discontinuity, corresponds to the optimal segment boundaries B^* and the identity of the sub-path (mapping a segment in the input feature vector sequence), in terms of the template q_k in the y-axis, gives the optimal label to the corresponding input segment s_k. The one-pass DP solution also yields the optimal matching distortion D^* corresponding to this optimal path, which is a measure of the overall quantization distortion in the joint segmentation and quantization.

2.2.3.5 Phoneme Recognition and Phonetic Vocoders

A typical application of the above 'connected word recognition' is the continuous phone recognition problem, where the input speech (in the form of a feature vector

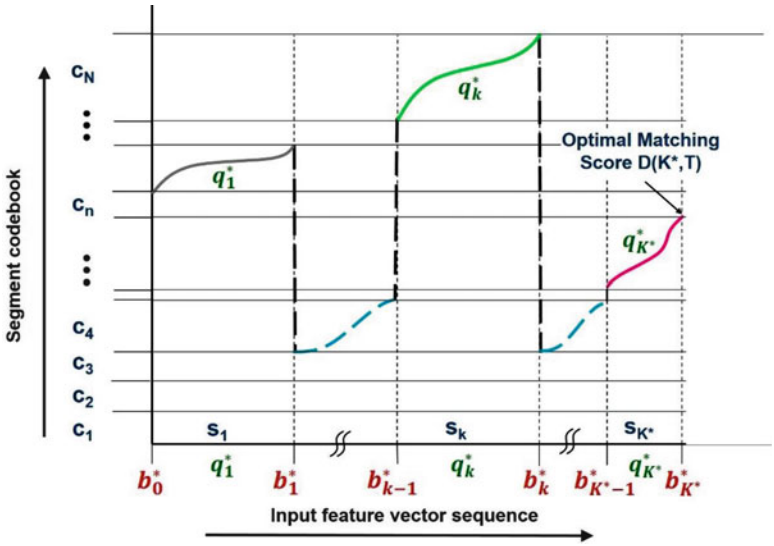


Fig. 2.14 Joint segmentation and quantization (segmentation and labeling) in an one-pass DP (Viterbi) framework

sequence) is decoded into a sequence of phonemes using a set of phone-models, typically HMMs, either in the form of monophone (context independent) models or triphone (context-dependent) models. This will be seen to be one of the primary joint segmentation and quantization technique repeatedly used in a good number of segment vocoders to derive phone-like segmentation and transmission of phonetic indices.

Though we attributed the advent of joint segmentation and quantization framework to Shiraki and Honda's variable length segment quantization algorithm and the joint segmentation and clustering algorithm for design of variable length segment codebooks, the principle of performing a 'connected phoneme recognition' using phone HMMs dates back to the work of [MCRK77, SKMKZ79, SKMS80] from BBN. These work represent the earliest work in segment vocoder framework focused on deriving and transmitting phonetic units at the encoder, using a phoneme recognition system and fall within the framework of joint segmentation and quantization. In this work, the transmitted phone indices were then used in the decoder to derive a diphone sequence that were used for synthesis, following an earlier feasibility study that established the merit of diphone synthesis that alleviates concatenation issues that is typical of phone-based concatenation and synthesis.

The next work (and an early one representing a conceptual milestone in ultra low bit-rate coding), that also employed phone recognition in the encoder (as in [SKMS80] referred above) and thereby falls within the joint segmentation-quantization framework is the 'phonetic vocoder' system of [PD89]. Here, the

input speech is segmented and labeled in terms of phonetic units using speech recognition techniques (essentially a connected phone recognition system) with the phone HMM acoustic models (numbering 60) trained from phonetically labeled TIMIT database. The state sequence path, as a sequence of state transition information, through each phone HMM in the resulting phone transcription is further used to specify the frame-level quantization of the input feature vector (10th order LPC coefficients), which are transmitted and used at the decoder to reconstruct the speech signal.

To generalize further, an entire class of segment vocoders evolved based on the principle of phone recognition using HMM based acoustic modeling of phonetic or acoustic sub-word units, and using conventional speech recognition techniques (for deriving a phonetic transcription of the input speech to be coded) [IP97, OPT00, T98, H03, MTK98, McC06]. All of these undoubtedly fall within the joint segmentation and quantization framework, by virtue of performing a phoneme recognition, implying a simultaneous solution to both the segmentation and labeling problem as defined above. Having noted this class of segment vocoders, we shall discuss the HMM based approaches separately in some detail in a section to follow (Sect. 2.4).

2.2.4 Segment Codebook

As a precursor to discussing variable length segment codebooks, it is best to consider the fixed length variants, namely, the vector quantization (VQ) codebooks and matrix quantization (MQ) codebooks. As the name implies, VQ is a codebook of single vectors, derived by the K-means algorithm or the LBG algorithm [M85]. MQ is derived similarly, by treating each matrix segment as an entity and defining matrix-to-matrix distortion, and setting the problem of MQ codebook design also in the K-means framework [M85]. The interesting work of Svendsen [S94], who showed that a ML segmentation can be made to yield a variable length segmentation, with each segment quantized by a single vector from a VQ codebook also fits into the notion of a vector codebook, but applied on variable length segments. In a similar vein, the R/D optimal linear prediction, which uses a collection of LP models (of various orders and quantized and coded in different ways) to quantize variable length segments, also fits into the notion of a vector codebook, though derived in a manner similar to variable-length segment quantizer design, but with additional constraints on the rate of the resulting codebook, which we shall show later (in a separate section devoted to R/D optimal LP coding, Sect. 2.3).

Segment codebooks that have been employed in segment vocoders typically fall in two categories:

(a) Templates: Variable length segments, also called templates, made of a sequence of feature vectors (e.g. LP parameter vectors, MFCCs, etc.) representing phones, diphones, syllables or automatically derived acoustic units.

(b) Hidden Markov models (HMMs): Parametric models of phones (context-independent monophones or context-dependent triphones), diphones, syllables or automatically derived acoustic units.

These type of segment codebooks have been used both for an independent segment quantization (following a first stage automatic segmentation) as well as in a joint segmentation and quantization step (as is more common for HMM based modeling of phonetic units in the form of phoneme recognition).

2.2.4.1 Template Segment Codebooks

The segment codebooks, in the form of templates are either derived as randomly populated segments (selected from a long sequence of speech feature vectors by means of automatic segmentation, such as described in Sect. 2.2.1 [RSM82b, RSM83, RWR87, RS04]) or designed by means of the joint segmentation and clustering algorithm (or the segmental K-means algorithm) as in [SH88], as pointed out earlier in Sect. 2.2.1. Considering the importance of the VLSQ scheme using the clustered codebook design, we add some details on this here. Figure 2.15 shows a typical joint quantization and clustering algorithm (also referred to as segmental K-means algorithm), which shares it iterative framework with the more conventional K-means algorithm, but now adapted to segments occurring in continuous speech, with the added step of having to extract them optimally in the first place, followed by defining a cluster of such segments and its centroid. This algorithm is illustrated in Fig. 2.15 and explained in the following.

The joint segmentation and clustering algorithm (or the SKM algorithm) comprises the following two steps carried out iteratively, until convergence determined typically by a rate of decrease in the overall quantization distortion over the training corpus:

1. **Joint segmentation and quantization**: This step first performs a segmentation and labeling of the input speech (training corpus) using an initial segment codebook, which could be randomly populated fixed length segments or variable length segments derived from some other automatic segmentation methods. This segmentation and labeling is carried out by the joint-segmentation and quantization algorithm (realized in the form of a 2-level DP algorithm, as noted in Sect. 2.2.3.3) as shown in Fig. 2.15a in the form of the input feature vector sequence being segmented and labeled in terms of the segment codebook indices, for e.g. the variable length segments (A, B, C, D, E, F, G) labeled with the index 32 are shown. For a segment codebook of size N, this step results in N clusters of variable length segments each indexed by one of the codebook segments. Each such cluster has a number of variable length segments which can be said to share the same acoustic property as the codebook segment whose index it is labeled by.
2. **Segment codebook update**: In this step, each codebook segment (corresponding to a cluster derived above) is replaced by the centroid (or also called the pseudo-

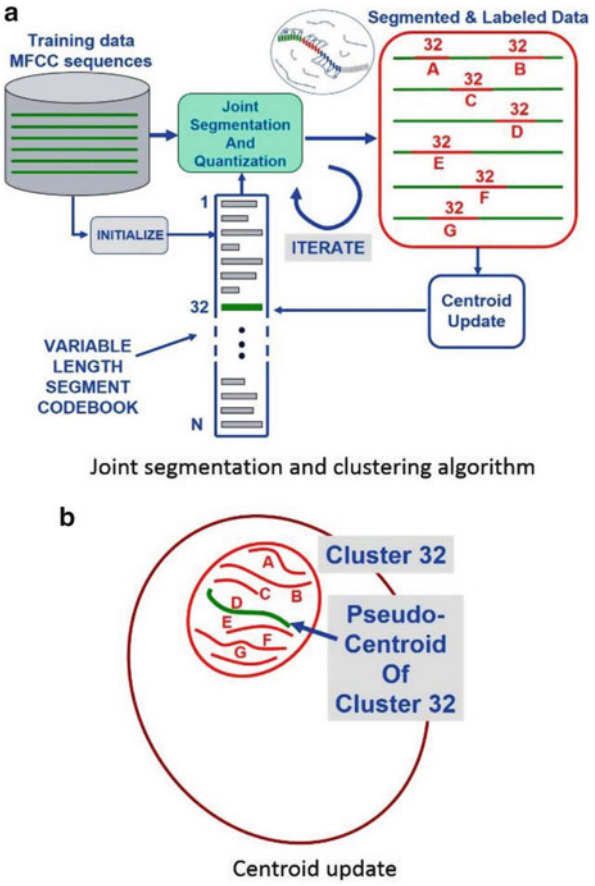


Fig. 2.15 Joint segmentation and clustering algorithm (or the segmental K-means algorithm). (**a**) Basic iterative structure, (**b**) Centroid update

centroid) of the cluster, defined as the segment (from among the segments in the cluster) which has the lowest average segment-to-segment distortion with all the other segments in the cluster. When defined as a pseudo-centroid, the resulting optimal segment satisfying this condition happens to be one of the segments in the cluster. This is illustrated in Fig. 2.15b, which shows a cluster corresponding to index 32, made of the variable length segments (A, B, C, D, E, F, G). The centroid update step chooses segment D as the pseudo-centroid (that minimizes the average intra-cluster quantization distortion) as the one that replaces the previous codebook segment 32. Once all the N codebook segments are updated in this manner, the new codebook is used to perform a variable length segmentation in Step 1.

The above two steps are carried out iteratively, until the average segment quantization distortion (for instance, as is obtained during the centroid update step, as the average of the N intra-cluster segment quantization distortion with

respect to the updated codebook) converges, i.e., the rate of decrease of this segment quantization distortion (on the training corpus) converges below a threshold. The segment codebook at the end of the convergence is the desired variable length segment codebook.

2.2.4.2 HMM Segment Codebook

The segment codebooks, in the form of HMMs (say, phone HMMs) are derived by now well established techniques in speech recognition, namely, acoustic model training. Here, the HMMs are trained from a training corpus (which could be manually or semi-automatically) labeled in terms of the units of interest (phones, diphones, syllables, etc.), e.g. as in [T98, H03, MTK98, C08, CMC08a, CMC08b, PSVH10]. Alternately, and more commonly, the training speech corpus would have only the associated orthographic transcription from which model-estimation algorithms such as 'embedded re-estimation' are used to train the HMMs. As will be indicated in a following section on HMM based recognition-synthesis techniques, an interesting departure from phone HMMs are the HMM modeling of abstract acoustic units (automatically derived units) corresponding to multi-grams which can span several successive phonetic units [CBC98a] or the recognition acoustic units (RAUs) and synthesis acoustic units (SAU) of [BC03].

2.2.5 Duration Modification

As noted in Fig. 2.5, the decoder receives the indices of segments (from the segment codebook) which quantize successive segments in the input (derived by either separate automatic segmentation and segment quantization or a single-step joint segmentation-quantization). These indices are used to retrieve the corresponding segments from a segment codebook, which is usually identical to the one in the encoder or identical in the nature of the underlying speech each segment represents or models (in the case of HMM), but represented using different parametric representation more conducive for synthesis. The retrieved segments are concatenated together and used for synthesis along with the residual information (also received by the decoder in some parameterized and quantized form) in the case of a LPC synthesis framework or other prosodic information in the case of alternate synthesis framework (such as HNM, PSOLA or HTS).

In the case of template based segment codebook representation, an important step in the decoder 'prior' to the concatenation of the templates corresponding to the received indices is 'duration modification' (also called 'duration normalization'), where a template in the segment codebook corresponding to a received index is modified to have a duration of the original segment in the input speech (at the encoder). The durational information is also quantized and transmitted to the decoder, thereby enabling this duration modification step. This has been typically

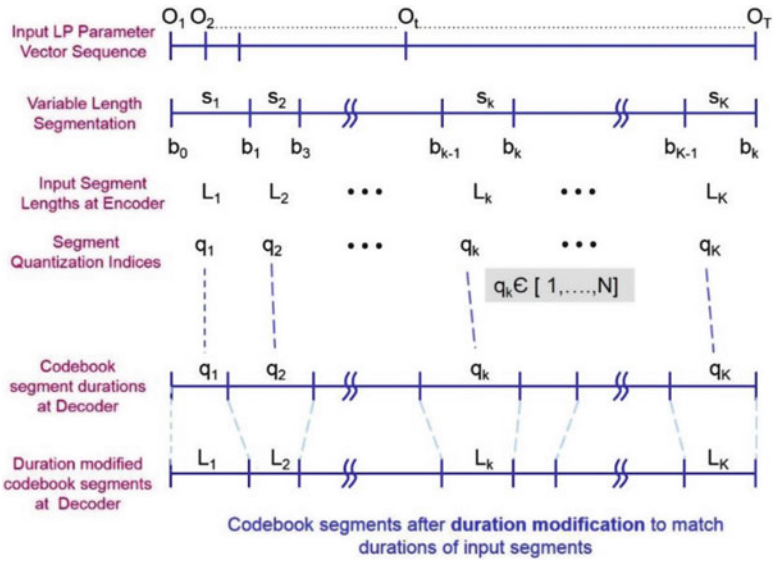


Fig. 2.16 Illustration of duration modification in the flow of processing starting from segmentation and quantization (at encoder) and duration modification codebook segments (at decoder)

done via space-sampling [RSM82b, RSM83, RWR87] (as depicted in Sect. 2.2.2) or by a linear warping matrix [SH88] or DTW warping path [CBC00]. In the case of the segment codebook being a HMM codebook, the duration modification is implicitly carried out during HMM-based synthesis using the state-duration information transmitted as side-information [PD89, T98, H03, MTK98].

The general notion of duration modification is shown in Fig. 2.16. The input LP parameter vector sequence $\boldsymbol{o}_1, \boldsymbol{o}_2, \ldots, \boldsymbol{o}_T$ is segmented into K segments $(s_1, s_2, \ldots, s_{k-1}, s_k, \ldots, s_K)$ of duration $(L_1, L_2, \ldots, L_{k-1}, L_k, \ldots, L_K)$ and corresponding segment quantization labels (or indices) $(q_1, q_2, \ldots, q_{k-1}, q_k, \ldots, q_K)$. The encoder transmits both the segment indices $(q_1, q_2, \ldots, q_{k-1}, q_k, \ldots, q_K)$ and the corresponding original segment durations $(L_1, L_2, \ldots, L_{k-1}, L_k, \ldots, L_K)$. At the decoder, let the segment codebook of size N (identical to the one at the encoder) be $C = (c_1, c_2, \ldots, c_n \ldots, c_N)$ with corresponding lengths $(l_1, l_2, \ldots, l_n \ldots, l_N)$. Each of the segments in the segment codebook which is retrieved using the received indices thus has its own duration, i.e., a segment in the codebook c_{q_k} retrieved using index q_k is supposed to represent the input segment s_k; the length of input segment s_k is L_k, whereas the length of the corresponding codebook segment c_{q_k} is l_k. This necessitates the duration modification of the codebook segment c_{q_k} (of duration l_k) to duration L_k so as to match the duration of the input segment s_k. Let this duration modified codebook segment be c'_{q_k}; the sequence of duration modified codebook segments at the decoder is now $\left(c'_{q_1}, c'_{q_2}, \ldots, c'_{q_{k-1}}, c'_{q_k}, \ldots, c'_{q_K}\right)$ and speech

synthesized from these concatenated units will have the individual units matching the corresponding units in input speech and together match the duration of the original speech.

2.2.6 Residual Parameterization and Quantization

Figure 2.5 showed the general structure of a segment vocoder in the framework of LP analysis and synthesis. In this framework, the spectral quantization part enjoys most attention given the importance of minimizing spectral distortion for high quality synthesis at the decoder. However, the role of the residual information in this LP framework is equally important, as is evidenced by the fact that the original LPC-10 vocoder is acknowledged to have a synthetic voice owing to the use of a voiced/unvoiced modeling of the residual in terms of a pulse train or random noise as the excitation signal as an approximation of the original residual in synthesis at the decoder. Much effort has been expended in rendering the LPC-10 quality better by an enhanced modeling of the residual, such as in the MELP coder [M08].

In the context of the segment vocoder framework shown in Fig. 2.5, the residual therefore continues to play an important role in the overall quality of synthesized speech at the decoder. A wide variety of residual modeling, parameterization and quantization has been adapted in the segment vocoder literature, each motivated by both parsimonious representation of the residual and ensuring minimal loss of speech quality during synthesis.

The basic modeling and representation of the residual in terms of voicing/unvoiced decision and pitch (with the gain parameter considered here as a global parameter, governing the short-term energy of the signal) continues to be the primary means of quantization in early segment vocoders, though in each case employing some special means of lowering the effective rate for the residual (e.g. 800 bps VQ-LPC uses 8 bits/frame for pitch, gain and voicing quantization by combining three consecutive frames in comparison to 13 bits/frame in the 2,400 bps LPC-10 vocoder). Specific techniques for incurring highly reduced bit rates for pitch or gain information typically includes modeling and quantizing pitch and gain profiles over an entire segment (the segments determined by automatic segmentation or joint segmentation quantization step) or use of vector quantization to quantize such pitch and gain vectors. For instance, we note here some salient techniques used in the residual (for LP based synthesis) or prosody (for other types of synthesis, e.g. HNM) modeling and quantization. In the earliest segment vocoder work, [RSM82b, RSM83] transmitted pitch with only 3 bits/segment and 1 bit/segment respectively, using a piece-wise linear approximation of pitch, with the pitch profile in a segment being modeled by a linear function, and the change in pitch quantized by an adaptive 2-level quantizer, with the level being proportional to the segment duration. The latter work on variable length segment quantization [SH88] used 4 bits/segment for differential pitch quantization.

Considering the more recent HMM based recognition-synthesis system, [T98] used pulse train or white noise excitation (of the MLSA filters) as in the classic LP vocoder, but reported results without quantization of pitch. In a subsequent work in the same framework, [H03] used a $F0$ quantization scheme based on a VQ version of the multi-space distribution HMM. Piece-wise linear approximation of the contour was adapted in other work too, such as in [LC99], in a rate-distortion framework for coding the pitch contours and notably in their later unit-selection framework [LC01, LC02], though used in a HNM based synthesis at the decoder. From among the series of work reported based on the ALISP units based on HMM recognition and synthesis by LPC or HNM, [PCB04a, PCB04b] can be seen to use unit selection based on a pitch profile correlation between the input segment and codebook segments, and quantizing a pitch correction parameter to apply on the codebook segment to match with the input segments pitch profile. In the parallel formant synthesizer used in the decoder in [OPT00], an appropriate excitation signal is chosen, depending on the amount of voicing and frication present in the sound being synthesized. [McC06] explored an interesting combination of the HMM based phonetic recognition and MELP based residual modeling within a predictive vector quantization of the MELP parameters for synthesis to realize a scalable phonetic vocoder framework operating at bit rates from 300–1100 bps.

2.2.7 Synthesis

As indicated in Fig. 2.5, the segment vocoder is typically set in LPC vocoder framework based on the source-filter model, with LP analysis in the encoder computing the spectral parameters (filter part) and the residual (source part) and quantizing and transmitting them. The synthesizer at the decoder uses the LPC synthesis framework to use the received filter and residual parameters to synthesize speech. When the residual is modeled by voicing decision, pitch and gain, the synthesized speech resembles that of LPC-10 vocoder. This is more or less the synthesizer framework and methodology adapted in a range of segment vocoders [W82, RSM82a, RSM82b, RSM83, RWR87].

A major departure from this was adapted in the unit-selection based coders of Lee and Cox [LC01, LC02], who employed the ‘harmonic plus noise model’ (HNM) framework, though continuing to use prosodic parameter estimation at the encoder, and their use in modifying the synthesis unit’s representation for HNM based synthesis at the decoder. The ALISP unit based work (reported in a series of papers which are reviewed in Sect. 2.5) [MBCC01, MGC01], explored both LPC synthesis framework and the HNM framework, choosing to prefer HNM over LPC synthesis, owing to the artifacts observed in LPC synthesis.

In an interesting departure from these work, [OPT00] used a parallel formant synthesizer at the decoder, driven by synthesis parameters derived from rules which in turn were acquired by a mapping from acoustic segments obtained by the speech recognition based segmentation at the encoder. In what can be considered as a

convergence of recognition-synthesis frameworks, the HMM recognition-synthesis work of [T98, H03, MTK98] used the HMM-based speech synthesis framework, now widely popular, successful and referred to as HTS [T13], where the synthesis is done using the MLSA filter (derived from the mel-cepstral coefficients derived from the concatenated state sequence of the recognized HMM phone units) and driven by a pulse train or white noise depending on the voicing decision. The fact that the quality of the HTS synthesis in a speaker-adaptive mode could reach a MOS of 4 [MTK98] seems to herald a new direction in ultra low bit-rate coding, where high quality speech is possible in a speaker-adaptive (and hence speaker-independent) mode of operation.

2.3 R/D Optimal Linear Prediction

In this section, we review a class of segment quantization algorithms, namely by Prandoni and Vetterli [PGV97, PV00], Chou and Lookabaugh [CL94] and the closely related work of Baudoin et al. [BCC97], all set within a formulation of rate-distortion optimization, where the segmentation and quantization is performed with respect to constraints on the overall rate of the resulting quantization, as well as impacting the underlying codebook design. We also provide an interpretation of Svendsen [S94], which sets a constraint on the distortion in performing the segmentation.

2.3.1 Prandoni and Vetterli R/D Optimal Linear Prediction

An important development in the framework of segment quantization is that of Prandoni et al. [PGV97] and Prandoni and Vetterli [PV00] termed R/D optimal linear prediction. To state briefly, in this work, segmental distortion is defined with respect to the LPC residual error and the bit-rate is defined as the trading-off parameter based on the cost of the LPC order for each segment, so as to yield a R/D trade-off in controlling the choice of LP order per segment and the resulting overall distortion, resulting in ultra low rate coding (working in a variable bit-rate manner) with average bit-rates of 300–900 bps.

Considering the importance of such a formulation in the segment vocoder framework, we give below in some detail the basic formulation of this work, but using the notations already introduced above in the definition of the variable length segment quantization framework in Sect. 2.2.3. A codebook is given in the form of a collection of N different LP models (which could be predictors of different orders as well as predictors whose parameters are quantized and coded in different ways). Let this codebook be $C = (c_1, c_2, \ldots, c_n \ldots, c_N)$. Let a segmentation and quantization of an input speech LP parameter sequence $O = (\boldsymbol{o}_1, \boldsymbol{o}_2, \ldots, \boldsymbol{o}_T)$ be given by the segment boundaries $B = (b_0, b_1, \ldots, b_{k-1}, b_k, \ldots, b_K)$, with corresponding segments

$(s_1, s_2, \ldots, s_{k-1}, s_k, \ldots, s_K)$ and the quantization labels $Q = (q_1, q_2, \ldots, q_{k-1}, q_k, \ldots, q_K)$. Here, the quantization label q_k represents a LP model c_{q_k} (of some order) from the codebook C. The distortion corresponding to such a segmentation and quantization is given by $D(O, K, B, Q)$ given by the sum of the K segmental LP prediction errors $d^2\left(s_k; c_{q_k}\right)$, where each of these errors is the squared LP prediction error when model c_{q_k} is applied on segment s_k, i.e.,

$$D(O, K, B, Q) = \sum_{k=1}^{K} d^2\left(s_k; c_{q_k}\right)$$

In addition, this work defines the overall bit-rate for such a quantization by using a cost function $r\left(c_{q_k}\right)$ that reflects the cost (in bits) as a function of the order $b\left(c_{q_k}\right)$, including the side information for specifying the segment duration and the relative LP order; i.e., the overall bit-rate load for a typical segmentation of O in terms of (K, B, Q) is given by

$$R(O, K, B, Q) = \sum_{k=1}^{K} r\left(c_{q_k}\right)$$

The important formulation in this work arises from solving the segmentation and quantization problem to yield the optimal (K^*, B^*, Q^*) under a rate-distortion trade-off specified by the constrained minimization for a given K, given by

$$(B^*, Q^*) = \min_{B} \min_{Q} D(O, K, B, Q) \tag{2.1}$$

$$R(O, K, B^*, Q^*) \leq R^* \tag{2.2}$$

where Eq. (2.2) dictates that the overall quantization distortion $D(O, K, B, Q)$ be minimized under the constraint that the overall bit-rate load be less than a specified bit-rate budget (or limit) R^*. This formulation is easily understood by noting that the segmentation and quantization solution of Eq. (2.1) for a given K is simply the ML kind of segmentation with the intra-segment distortion of a segment being replaced by the squared LP error with respect to the 'best' LP model from the collection of possible LP models C. As with the ML formulation, this overall distortion is monotonically decreasing for increase in K, as the segments becomes shorter and shorter. However, when it is required that the overall bit-rate be limited to a maximum of R^*, this translates into the maximum number of segments that can be derived and further to the maximum order of the LP model that can be used for quantizing a segment; i.e., without the bit-rate budget constraint, an extreme solution is one where the number of segments equals the number of frames $(K = T)$, i.e., each frame is modeled by its best LP model from the collection, which in turn can be the LP model of the highest order available. However, with the bit-rate budget in place, the number of segments is optimized with the bit-rate

resource distributed over the segments, so that each segment is modeled by some LP model of lower order (not necessarily the highest order among the models available). This can also be seen as a combination of two functions as a function of the number of segments K, one being the overall distortion which decreases with K and the overall bit-rate which increases linearly with K.

When the constrained minimization problem in Eqs. (2.1) and (2.2) is combined via a Langrangian multiplier, these two functions combine to yield an optimum (minimum) for some K^*. Such a joint functional using a Lagrange multiplier is given by,

$$J(\lambda) = D(O, K, B, Q) + \lambda R(O, K, B, Q)$$

which can further be simplified to yield the optimization as

$$J^*(\lambda) = \min_{B} \min_{Q} \left\{ \sum_{k=1}^{K} d^2\left(s_k; c_{q_k}\right) + \lambda r\left(c_{q_k}\right) \right\}$$

It can be easily seen that this is exactly in the form of a ML segmentation that can be reformulated and solved by the 2-level DP algorithm outlined above in Sect. 2.2.3.3, by defining the distortion (or cost) associated with the kth segment [in the recursion of Eq. (2.3)] as the term

$$\min_{c_{q_k} \in C} \left\{ d^2\left(s_k; c_{q_k}\right) + \lambda r\left(c_{q_k}\right) \right\}$$

for some choice of λ (which weighs the bit-rate load in addition to the distortion function appropriately). The recursion, as in the 2-level DP algorithm, is given by

$$J^*_{[1,b_k]} = \min_{1 \le b_{k-1} \le b_k - 1} \left\{ J^*_{[1,b_{k-1}]} + \min_{c_{q_k} \in C} \left\{ d^2\left(s_k; c_{q_k}\right) + \lambda r\left(c_{q_k}\right) \right\} \right\} \tag{2.3}$$

where $J^*_{[1,b_k]}$ is a k segment segmentation of the input segment $O_1^{b_k} = \{\boldsymbol{o}_1, \ldots, \boldsymbol{o}_{b_k}\}$ described recursively as the sum of the optimal $k-1$ segmentation of the segment $O_1^{b_{k-1}} = \{\boldsymbol{o}_1, \ldots, \boldsymbol{o}_{b_{k-1}}\}$ (i.e., $J^*_{[1,b_{k-1}]}$) and the minimum cost (segmental squared LP error of segment $s_{\boldsymbol{k}} = O_{b_{k-1}+1}^{b_k} = \{\boldsymbol{o}_{b_{k-1}+1}, \ldots, \boldsymbol{o}_{b_k}\}$ plus the corresponding λ weighted bit-rate load) associated with segment s_k. The final solution is obtained by invoking the above recursion [Eq. (2.3)] for $J^*_{[1,N]}$ for a given λ and finding the optimum over a desired range of K. Note that the solution is obtained in the same manner as the 2-pass DP kind of recursion solved by a trellis realization, and recovering the optimal segmentation and quantization parameters by backtracking.

2.3.2 *Variable-to-Variable Length Vector Quantization*

In the context of designing segment codebooks within the constraint of a rate-distortion trade-off, a related work is that of Chou and Lookabaugh [CL94]. This is an important work in the following ways: It formulated the problem of 'variable-to-variable' length codes, where the terminology derives from whether the segments (or blocks) in the input speech (parameter sequence) vary in length or not and whether the blocks of channel symbols vary in length or not. The term 'variable-to-variable' refers to a scheme that quantizes variable length segments into variable length bit patterns (channel symbol). While it is interesting to note that this formulation handles variable length segments in the input speech, the 'variable' part of the channel symbol comes from performing an entropy coding (e.g. Huffmann code) that exploits the non-uniform probability distribution of the segment usage from a segment codebook. While this formulation is set in a constrained rate-distortion optimization (as in [PV00] above), the solution derived is very similar to the one by [PV00] in the form of a recursive relation that can be seen as a variant of the ML-formulation (Sect. 2.2.1.2) redefined into the 2-level DP solution. [CL94] also give an iterative solution for the design of the segment codebook, along the same lines as the joint segmentation and clustering (but with the additional bit-rate constraint on the joint segmentation and quantization step) or the segmental K-means algorithm as noted above in Sect. 2.2.4.1.

An important aspect in this formulation is that, though the segment codebook is made of variable length segments, the mapping between an input segment to the codebook segment that quantizes the input segment is such that they are of same length, i.e., it does not perform any kind of segment to segment warping between potentially unrestricted set of variable length codebook segments to each input segment as may be required. [CL94] points to this as an important difference from Shiraki and Honda's [SH88] variable length segment quantization algorithm. As a result of not using segment to segment warping, [CL94] show that their method outperforms that of [SH88]. Moreover, [CL94] note that a related work Jeanrenaud and Peterson [JP91, PJV90] from BBN, is more close to their formulation, in the sense that [JP91] use duration dependent segment codebooks, which is populated (and designed by means of an iterative algorithm) with segments of variable lengths, but ensuring that it has sub-codebooks each with several segments of a fixed length. However, it should be noted that having a variable length segment codebook and allowing for warping based segment quantization allows a particular segment representing a particular acoustic realization of speech to quantize a varied number of input segments belonging to the same acoustic category; in the absence of such a warping of a particular segment in the codebook, the codebook is constrained to have segments of different lengths to match with the different realizations of the same acoustic category (i.e. without warping), thereby increasing the effective codebook size and the bit-rate to achieve a desired quantization distortion.

2.3.3 Multigrams Quantization

In a related work, Baudoin et al. [BCC97] provided a reinterpretation of the VVVQ method of Chou and Lookabaugh [CL94] with reference to the multigrams quantization method, where a spectral vector sequence is segmented and quantized using a codebook of variable length segments called multigrams, and the codebook is obtained by maximizing the joint likelihood of the optimal segmentation and the observation sequence under two different scenarios, one when in the form of quantization indices (classical multigrams) and the other without vector quantization (modified multigrams) [BCC97]. The concept of multigrams was earlier introduced in an interesting formulation by Bimbot et al. [B95], which allowed segmentation of a symbol string (e.g. text data or spectral sequence indices resulting from VQ) into characteristic sequences, and building a dictionary of such sequences using a maximum likelihood parsing of a string W using Viterbi decoding, given by

$$L(W) = \max_{\{B\}} \prod_{k} p(S_k)$$

where, B is the set of all possible segmentations into variable length sequences S_k. The solution to this parsing is expressed in a recursive manner akin to the maximum likelihood (or the 2-level DP algorithm) formulation and solved by dynamic programming.

2.3.4 Distortion Constrained Segmentation

In the context of the above R/D optimal segmentation, it can be noted that the earlier work of Svendsen [S94] can be restated in the form of R/D constrained segmentation, but with the difference that in this work, the segmentation was optimized to let the distortion be not above a threshold distortion (corresponding to the 1 dB transparent quality quantization), and the associated segmentation was observed to reduce the bit-rate by a factor of 2, i.e.., the number of segments could be reduced to the extent that each segment can be made of several frames thereby ensuring more effective use of a codevector, and such that when each segment is quantized by the best codevector in a vector quantization codebook, the average distortion is limited to be less than the specified threshold. This is illustrated in Fig. 2.17. Note that point A corresponds to the baseline case of each frame being quantized by a codevector in the VQ codebook; the corresponding distortion is not 0, as was the case with ML segmentation (for the limiting case when the number of segments K equals the number of frames T in the input sequence and where the centroid happened to be the approximating vector), since here the quantization of the individual frames is done with an external vector codebook

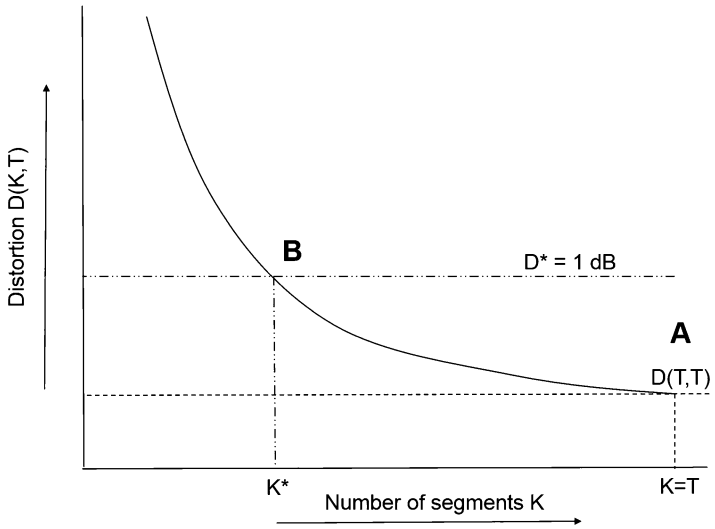


Fig. 2.17 Distortion vs. number of segments in the distortion constrained segmentation (in ML segmentation formulation) of [S94]

thereby yielding a minimum non-zero distortion, which sets the asymptotic performance, when $K = T$. Point B corresponds to the solution obtained by the algorithm of Svendsen [S94] characterized by an average distortion not greater than the specified threshold and the corresponding bit-rate derived from the associated number of segments (effective bit-rate = segment-rate $K^*/\left(\frac{T}{F}\right)$ segments/second $\times \log_2 N$ bits/segment) for a VQ codebook of size N and a frame-rate of F frames/second.

The ML formulation of [SS87], adopted in [S94] as above in a distortion-constrained segmentation can also be seen to generalize over the combined quantization-interpolation (CQI) [LF93], referred to in Sect. 2.1, by noting that the intra-segmental distortion of a segment (as defined in the sub-section on ML-segmentation in Sect. 2.2.1.2) needs to be defined as the distortion due to approximating the frames in the segment by interpolated and quantized frames using the end frames of the segment as the 'target' vectors as defined in [LF93], even while retaining the constraint of realizing a distortion within some predefined threshold.

2.4 HMM Based Recognition-Synthesis Paradigm

As noted already in Sects. 2.2.3 and 2.2.4, joint segment quantization using phone-HMM codebooks for realizing a phonetic transcription of input speech represents a major departure in segment vocoder framework from the more classical segment

codebooks in which each segment is a template comprising a sequence of LP parameter vectors and an associated template based segment quantization scheme. Following the early work of [SKMS80, PD89], the use of HMM codebooks and phonetic transcription has further lead to a class of segment vocoder systems that have refined the HMM based approach into a recognition/synthesis framework and have resulted in very high quality speech coding. We briefly review these here.

2.4.1 HTS Based Framework

Subsequent to the above early work that explored phonetic HMMs in a phonetic vocoder setting, a major initiative in HMM based segment vocoders emerged with the very successful paradigm of parametric speech synthesis, or HMM-based synthesis, now referred to as HTS [T13]. We review these briefly here [T98, H03, MTK98] which were originally referred to as being set in a 'recognition-synthesis' paradigm. Figure 2.18 shows a generic structure of the HMM based recognition-synthesis framework adapted in these work.

Tokuda et al. [T98] proposed an HMM-based speech recognition and synthesis technique for a very low bit rate speech coder operating at 150 bps and yielding subjective quality comparable to that of a VQ at 400 bps (50 frames/s and 8 bits/frame) and a DMOS of 3.4. This system derives the name of being in 'recognition-synthesis' paradigm, as the encoder is a HMM based phoneme recognizer and the decoder does the inverse operation of using an HMM-based speech synthesis

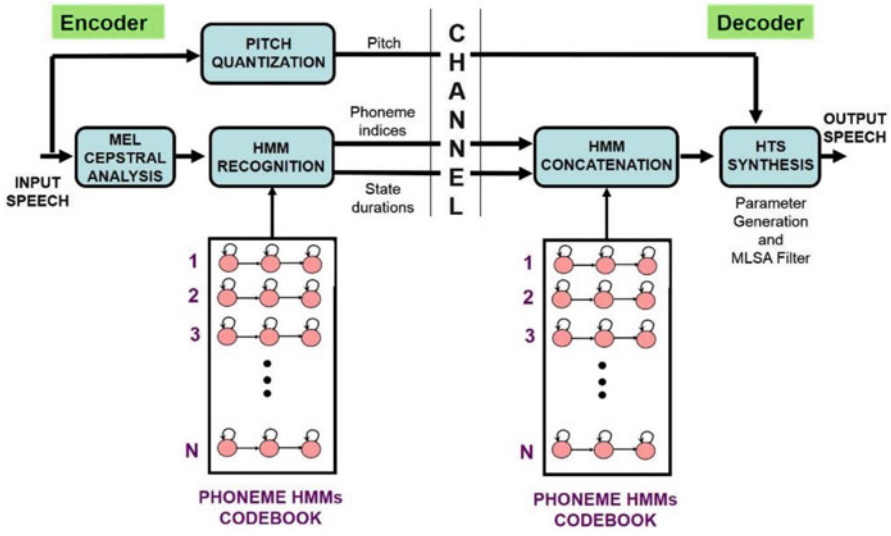


Fig. 2.18 Generic structure of HMM based recognition/synthesis framework

technique. More specifically, here, the speech spectra is represented by mel-cepstral coefficient vectors and transcribed into a phoneme sequence using context dependent (triphone) phoneme HMMs, where each triphone HMM was a 3-state left-to-right model with no skip. A total of 34 phonemes and silent models were used. Subsequent to phoneme recognition at the encoder using these triphone HMMs, the phoneme indices are transmitted to the decoder along with the state duration and pitch information. In the decoder, the phoneme HMMs corresponding to the phoneme indices are extracted from the phoneme-HMM codebook and concatenated and a sequence of mel-cepstral coefficient vectors (conditioned on the state sequence reconstructed from the received state duration information) is generated using a 'speech parameter generation' algorithm first proposed for HMM based speech synthesis [T13]. An important consequence of the parameter generation result is that the mel-cepstral coefficient vectors thus obtained reflect not only the means of static and dynamic feature vectors but also their covariances, resulting in a natural-sounding synthetic speech. Subsequent to the derivation of the mel-cepstral coefficients from the concatenated HMM, speech is synthesized by exciting a MLSA (Mel Log Spectrum Approximation) filter (derived from the mel-cepstral coefficients) by a pulse train or white noise generated according to the pitch information. However, a drawback of this coder was that it was speaker dependent.

Further improvements of the above phoneme-based HMM vocoder were proposed by Hoshiya et.al. [H03]. It was found that, by using a pitch coding scheme for quantizing each of the phoneme segments of the low bit-rate speech coder, the coder's performance at 110 bit/s was superior to that of a 600 bit/s VQ based vocoder in terms of the subjective MOS quality.

In a variant of these 'phonetic vocoder' approaches, [CCTC97] proposed a recognition-synthesis coder using HMM based phone modeling (48 phones and 2463 diphones) for phoneme recognition of the input speech to transmit the phonetic indices and further synthesis at the decoder using time-domain pitch-synchronous overlap method (TD-PSOLA) to realize a 750 bps coder which preserved speaker characteristics and had a MOS of 3.0.

2.4.2 Speaker Adaptive HMM Recognition-Synthesis

In order to render the first HMM-based recognition-synthesis vocoder speaker-independent, Masuko et al. extended the work in [T98] to be speaker adaptive in [MTK98]. The decoder was initially populated only with synthesis units from a single speaker. Therefore, regardless of the input speaker, the synthetic speech was limited only to the HMMs used in the decoder. Furthermore, it was likely that the mismatch between the training and input speakers caused recognition errors, further degrading the quality of the output speech quality. This problem could be handled in two ways. One was to have speaker dependent codebooks. The other more

reasonable approach was to adapt a standard codebook to the input speech by accounting for any mismatch between the input speech and the HMMs trained at the decoder. As a result, the HMMs at the decoder are adapted to the input speech by moving the output distributions of HMMs by adding a transfer vector to mean vectors to fit the distributions to the input parameter vectors. The transfer vector is determined for each phonetic segment by maximizing the output probability of the input sequence.

In this work, the segment codebook comprised 49 speaker independent, 5-state left-to-right triphone HMM. Transfer vectors are also trained for every 100 ms of speech contributing an additional bit-rate of 100 bps. The speaker dependent (SD) coder of [MTK98] was shown to significantly improve the quality of the coded speech and the recognizability of the speaker as opposed to the speaker independent (SI) system in [T98]. For one of the speakers in the system, a Degradation Mean Opinion Score (DMOS) of 3.8 was reported for the SD coder and 4.0 for the speaker adaptive system, while the DMOS for an SI coder was 1.5. This remarkable enhancement in DMOS for speaker adaptation can be considered as a milestone in ultra low bit-rate speech coding, given that the overall coder now becomes applicable to unseen test speaker, even while being capable of yielding high DMOS, in keeping with the underlying HMM-based synthesis system's potential to yield high quality synthetic speech, which currently marks the success of the HTS framework for TTS [T13].

2.4.3 Ergodic HMM Framework

In a related, but conceptually different, work within the HMM-based recognition-synthesis paradigm, [Lee05] proposed use of an ergodic HMM to replace the set of phone-HMMs, by having a large number of states (e.g. 64) to model any spoken language as a sequence of abstract acoustic units, which correspond to the individual states of such an ergodic HMM. By definition, such an ergodic HMM is trained from a long unlabeled training corpus and enjoys the advantage of not requiring phonetically labeled training corpus as is required for the phone-HMMs of [T98, H03, MTK98]. A transcription of input speech using such a large ergodic HMM yielded a state sequence which is transmitted to the decoder along with fractional pitch. The synthesis at the decoder follows the same HMM-based speech synthesis as in [T98], first by deriving spectral parameters from the HMM and the transmitted state sequence (MLSA filters derived from the mel-cepstral coefficients of the state-sequence) and mixed excitation signal from a MELP decoder using band-pass voicing strengths associated with a feature vector of the HMM to enable increased naturalness. While no direct comparison with the previously established HMM segment vocoders of [T98] or [MTK98] is done, this method shows that an ergodic HMM with number of states as 128 yields a good overall quality and intelligibility with speaker characteristics preserved at an effective bit-rate of 128 bps, though no

formal listening tests are done. The notions of the optimal number of states being 64 for American English to correspond to the phones in this work is based on earlier work [Pepp90, FC86, Pepp91] and is closely in corroboration with a related work on using ergodic-HMMs to model spoken languages for language-identification [RSS03, SR05].

2.4.4 Ismail and Ponting HMM Based Vocoders

In a work contemporary to that of [T98], Ismail and Ponting [IP97], Ismail [I98], and Ovens, Ponting and Turner [OPT00] propose a 300 bps HMM based vocoder also in a recognition-synthesis framework. This system is a variant of the methodology adapted in [T98] in at least 2 ways: Firstly, it uses a HMM that models abstract acoustic sub-word unit segments, rather than phones. In this context, it is also task-dependent and talker dependent and is also word-mediated in the sense that the HMM recognizer works with a vocabulary of 500 words, an associated pronunciation dictionary and acoustic models, and the recognized sub-word unit acoustic segment sequences are constrained to correspond to sequences of words from the known (500 word) vocabulary. Secondly, at the decoder, for synthesis it uses a parallel formant synthesizer, in tandem with a rule-based system that maps the recognized acoustic segments to formant parameters; i.e., it does not use HMM-based synthesis, quoting the trainable HMM synthesis framework of Donovan and Woodland [DW95] as computationally intensive for training for a single talker, though considering the system of [MTKI97] as viable for its rapid speaker adaptation ability.

In this system, the incoming speech is transcribed into a sequence of sub-word acoustic units with corresponding pitch and duration information. The sub-words are modelled around phones which are expanded to segments by defining the context and using a set of rules to define such an expansion. The HMMs modelled on such sub-word units allow for flexibility to extend the dictionary without having to train models afresh for new words. The pitch and voice information is also transmitted to add to the naturalness of the synthesized speech. This approach of HMM recognition followed by synthesis-by-rule where the synthesizer parameters are derived from natural speech results in a synthetic quality very nearly as the original speech. [OPT00] concludes that, though no formal evaluation of the system was performed, informal listener tests indicated that speaker-specific characteristics are preserved and that the re-synthesized speech sounds more natural than speech coded using a 2,400 bps LPC-based system.

2.4.5 *Formant Trajectory Model Based Recognition-Synthesis*

While the above sections dealt with various types of HMM based recognition-synthesis frameworks for ultra low bit-rate speech coding, we briefly describe the work of Wendy Holmes [H98] which is also set in the recognition-synthesis framework using HMMs, but with two important differences from the conventional HMM based recognition and synthesis, and being more in line with the work of Ismail and Ponting in Sect. 2.4.4 for the synthesis part: Firstly, the work in [H98] used a feature set comprising of the first three formant frequencies together with five mel-cestrum coefficients for a phone-like recognition using phone-level linear-trajectory segmental HMMs. The identified segments are represented by straight-line formant parameters for coding. Secondly, speech is synthesized at the receiver using a parallel-formant synthesizer (rather than HTS as in the systems described in Sects. 2.4.1, 2.4.2 and 2.4.3), driven by the frame-by-frame control parameters derived from the formant parameters.

The system was shown to produce speech with good intelligibility and preserving speaker characteristics at 600–1,000 bits/s. This work specifically highlights in being a unified model, in the sense of using the same linear formant-trajectory model for both recognition and system and further emphasizes that such a model used for recognition-based coding represents speech in such a manner that the model can be used for coding at a range of data rates, trading bits for a graded speech quality, incorporating increasing speaker characteristics at higher rates, over and above a baseline phoneme sequence synthesis (at the lower end of the data rate), though this paper focuses on the high bit-rate end of the range—i.e., coding formant trajectories—but not yet demonstrating the graceful degradations that the unified model is claimed to be capable of (for decreasing rates), which would have been a very appealing and dramatic phenomenological result of such a modeling approach, in the context of speech coding, and something yet to be attempted in most speech coding formulations.

2.5 ALISP Units and Refinements

In a significant departure from use of phone-like or diphone-like units to define the segmentation and labeling and the segment codebooks in most segment vocoder frameworks, a series of work reporting a coherent evolution of techniques first explored the use of ALISP units (Automatic Language Independent Speech Processing) [C98] that could span multiple phones as the coding units (for segmentation and labeling) and progressed towards different kind of recognition and synthesis units by refining the ALISP units in order to address various issues such as concatenation continuity, corpus based dynamic selection of units, speaker

adaptation, robustness to noise, etc.. In the following we give an overview of this series of contributions, highlighting the main aspects of each publication even while maintaining the chronological progression of techniques proposed and evaluated.

2.5.1 Basic ALISP Framework

Primarily based on the early work of Cernocky [C98], the ALISP units were motivated towards finding alternative approaches to defining units in contrast to the conventional notions of sub-word units derived based on phonetic knowledge. The ALISP units are defined using a combination of techniques—temporal decomposition, unsupervised clustering (e.g. vector quantization) of the target vectors so derived and multigrams, followed by HMM based modeling of the segments associated with multigram labels. One of the application of the ALISP units in [C98] was for very low bit-rate speech coding, realizing intelligible speech at an average bit-rate of 120 bps for two sets of speaker-dependent experiments.

The earliest of the work reported in this direction is [CBC97b] where the authors follow their early experiences with multigrams and modified multigrams [CBC97a] similar to the earlier VVVQ formulation of [CL94]. The main part of this work is on the lines of proposing the techniques further reported in [CBC98a, CBC98b, B99] which is described in some detail below.

This procedure first applies a temporal decomposition technique of the speech feature vector sequence, first proposed by [A83], to derive a series of spectral events each consisting of a target and an interpolation function, representing speech as being made of steady-state vectors blended by the interpolation function as would typify an underlying articulatory process. The parameter vectors (e.g. LPCC vectors in [CBC98b]) located at the gravity centers of the interpolation functions are vector quantized to obtain a string of symbols, from which a set of characteristic variable length symbol patterns called multi-grams (MG) are derived, further leading to a dictionary of multi-grams; in [CBC98a], a MG dictionary of size 1,666 is used, made of 64 1-g, 1,514 2-g and 88 3-g, with corresponding average length of a sequence in terms of spectral events being 1.638 of 112.7 ms. Each sequence in this MG dictionary is represented by a HMM trained on a training corpus labeled by the MG entries. This yields HMMs that model variable length units of speech, but with an underlying correspondence with speech units defined by the multi-gram process, rendered meaningful by the fact that each such sequence is derived and quantized by a temporal decomposition. The HMMs were left-to-right, with number of emitting states being proportional to the number of temporal-decomposition events in the modeled sequence. The HMMs are used as in phone transcription, but now yielding variable length acoustic segments matching the sequences that each HMM models. The work

reported natural synthetic speech, though with limited experiments, and pointed to various possibilities including different synthesis methods (PSOLA, MBROLA) and speaker adaptation.

Further, [CKBC99] combine the earlier reported approach of ALISP units (based on TD, VQ, and HMM) with syllable-segments which were derived from semi-automatic segmentation procedure used for a syllable speech synthesizer. The system was reported to operate at 175 bps on test data for ALISP derived units with a spectral distortion of 4.39 dB and having subjectively intelligible speech, though unnatural and with strongly audible artifacts. The syllable based approach had an average bit-rate of 62 bps with spectral distortion of 8.9 dB and subjectively poor quality of the synthesized speech which was hard to understand.

2.5.2 Re-segmented Long Synthesis Units

In a progression from the basic ALISP units, [CBC00] propose a technique at 120–195 bps with main emphasis on identifying the need to use a dichotomy of recognition (or coding) units (CU) and synthesis units (SU) and in going from phone-like units (defined from transition to transition, marked by intersections of the interpolation functions obtained from temporal decomposition) to long synthesis units that are defined from steady-state region to steady-state region, as marked by TD target vectors. Closely following this work, [MBCC01, MGC01] propose specific techniques for defining 'new units' that were derived from the 'original units' which were characterized by having unstable parts at their boundaries, contributing to 'transition noise' due to poor concatenation (high discontinuities). This paper further motivates the need to reduce these poor concatenation by proposing new units that can be obtained by a re-segmentation of the original units from stable part to stable part (steady-state to steady-state) so that the concatenation discontinuity is low. Specific techniques for such a re-segmentation is proposed, namely, according to middle frames of original units, or according to middle frames of middle states of original unit HMMs, or according to gravity centers of original TD-based units. This paper also marked a departure from LPC analysis-synthesis to HNM based synthesis considering the LPC framework to contribute to artifacts in the synthesized speech. This work also discusses the choice of 'synthesis units', for each coding unit wherein the best synthesis units (from three representatives) is chosen, using minimum DTW distance between a representative and an input speech segment, with the DTW path transmitted to the decoder. The paper reported a speaker dependent coder at 370 bps with a best spectral distortion of 5.5 dB.

http://www.springer.com/978-1-4939-1340-4

Ultra Low Bit-Rate Speech Coding
Ramasubramanian, V.; Doddala, H.
2015, VII, 152 p. 60 illus., 56 illus. in color., Softcover
ISBN: 978-1-4939-1340-4