Engineering Part IIB: Module 4F11
Speech and Language Processing
Lecture 14: Text-to-Speech Synthesis

Bill Byrne

Lent 2016

Cambridge University Engineering Department

What is Text to Speech Synthesis?

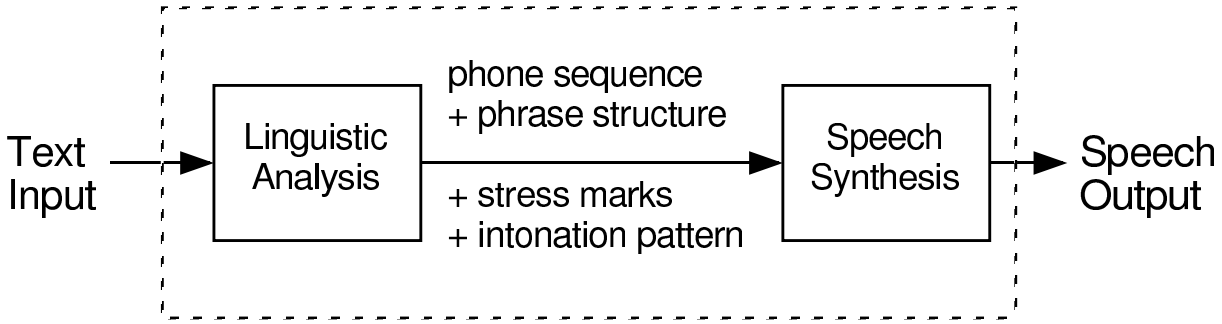
Speech output from systems is now common-place. However, the form of the input and the methods used for generating speech vary considerably:

1. reproduction of whole pre-recorded messages
2. concatenation of pre-recorded voice fragments
3. generation by rule of control parameters for a speech production model
4. concatenation of pre-stored basic speech sounds

Range of input and quality varies according to the scheme.

- 1 and 2 are limited by the choice of pre-recorded material. They typically produce high quality output.
- 3 and 4 allow arbitrary message generation, but quality is harder to maintain.

Text to Speech Synthesis (TTS) refers to systems which allow arbitrary input text (methods 3 or 4 above).



The stages of a TTS system.

As shown above, TTS is a two-stage process.

- **Linguistic analysis stage**: maps the input text into a standard form; determines the structure of the input, and finally decides how to pronounce it.
- **Synthesis stage**: converts the symbolic representation of what to say into an actual speech waveform.

- P. Taylor, "Text-to-Speech Synthesis". http://www.amazon.co.uk/Text-Speech-Synthesis-Paul-Taylor/dp/0521899273

Applications of TTS

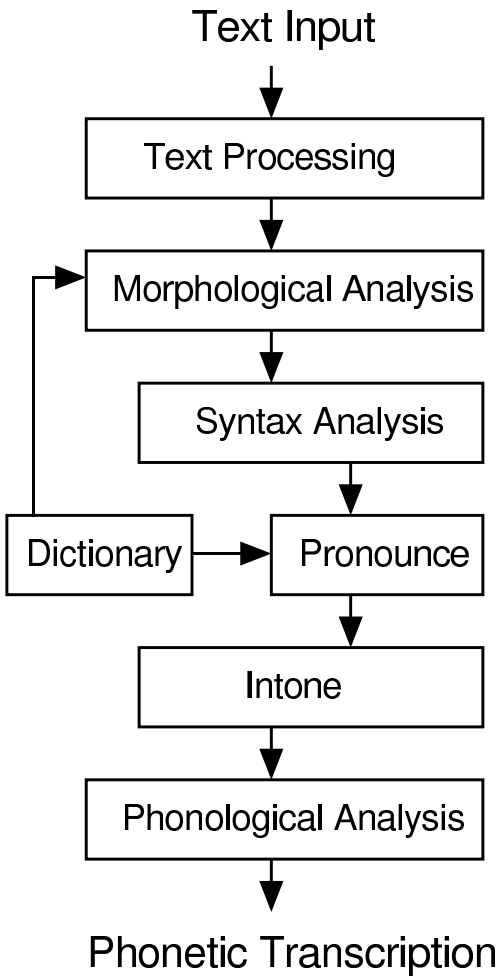
Text to speech systems are used in a wide range of application areas, e.g.

- proof-reading in word-processors
- language tutoring systems
- information access over telephone
- aid to the handicapped
- implementing interactive systems: e.g. games, simulators, toys

In addition, the linguistic analysis component can be useful in determining pronunciations for speech recognition systems, particularly names since these are not included in standard pronouncing dictionaries.

Linguistic Analysis

The major processing steps involved in linguistic analysis are shown below.



Record 2 songs

Record two songs

Record two song+s

(Verb)(num NPlural)

r ih k ao' d t uw s oh ng z

r ix kcl k ao' dcl t uw s oh ng z

Text Processing

This stage involves converting the unrestricted input into a standard form. The text processing stage appears at first sight to be trivial but in practice it is very hard to make it robust to arbitrary inputs.
Problems:

- **End of sentence markers** - important for synthesis. Many systems use end of sentence marker to also indicate, for example, an abbreviation.
- **Abbreviation expansion** - same abbreviation has multiple expansions determined by context. For example *St.* maps to *Saint* or *Street*.
- **Numbers** - pronunciation is dependent on meaning. For example whether it's a time, date, year, telephone number, or currency.
- **Specialist application** - e.g. expansion of chemical symbols, resistor values.

Examples:

£12.60	→	twelve pounds sixty
12.60	→	twelve point six
IBM	→	I. B. M.
"No comment"	→	quote No comment

Morphological Analysis

Most pronunciations are generated by looking up each word in a dictionary. However, words often have many variants and it is undesirable to have individual dictionary entries for each variant.

Morph decomposition:

- breaks words into *prefixes*, *roots* and *suffixes*.
- if root pronunciation is known, word pronunciation may be found by rule.
- assists the process of determining syntactic class.

Examples of morph decomposition:

re-programmable	$\rightarrow$	re + program + able
scarcity	$\rightarrow$	scarce + ity
timeout	$\rightarrow$	time + out

Morph decomposition is non-trivial. The basic approach is to recursively search each word for a covering decomposition in which all constituent morphs are in the dictionary. Morphs are tried from right to left, looking for the longest first.

To deal with spelling changes, the decomposition can include the application of one or more $pattern \rightarrow action$ rules. For example :

Pattern	Action	Example
xx + i	none	telling → tell + ing
	xx → x	running → run + ing
	xx → xxe	silhouetting → silhouette + ing
xx + ?	none	teller → tell + er
	xx → x	reddest → red + est

Note that the same pattern can trigger more than one action. Each must be tried in turn until the required covering decomposition is found. For example, the decomposition of "scarcity" would involve evaluating the following possibilities

scarcity → scar + city
→ scar + cite
→ scarce + ity
→ scarcity

Of these, the third is preferred because it has only a single root and it is a shorter root than the fourth (null) decomposition.

Syntax Analysis

The aims of the syntactic analysis phase are:

1. determine the lexical categories of each word e.g. "permit" can be a verb or a noun and it is pronounced differently in each case.
2. identify the phrase boundaries to assist in determining intonation

The techniques used are common to parsing in natural language understanding and compilers. The parser must handle arbitrary text but the detailed syntactic structure is not required.

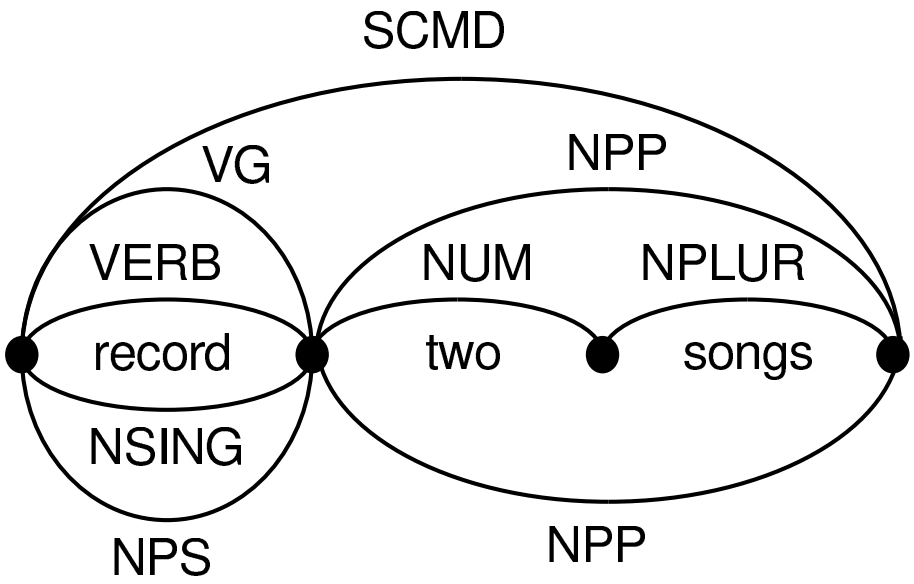
The morphological stage will have identified possible lexical categories from the dictionary, but some words may be ambiguous (e.g. noun or verb).

The parser uses rules of the following form to build arcs spanning one or more lower level arcs.

$$
\begin{array}{lcl}
\text{NPP} & \rightarrow & \{\text{NUM}|\text{ADJ}\}\ \text{NPLUR} \\
\text{NPS} & \rightarrow & \{\text{NUM}|\text{ADJ}\}\ \text{NSING} \\
\text{VG} & \rightarrow & \{\text{ADV}\}\ \{\text{AUX}\}\ \text{VERB} \\
\text{SCMD} & \rightarrow & \text{VG}\ [\text{NPS} \mid \text{NPP}]
\end{array}
$$

The braces denote zero or more repetitions, the square brackets denote options and the vertical bar separates alternatives.

In the example below, two different parses are possible but the interpretation of "record" as a verb is preferred since it leads to a completely spanning arc.



Parsing hierarchy

Pronunciation

The pronunciation component is responsible for converting words into phone symbols. Problems:

- Number of words is very large.
- Pronunciation for same letters varies according to context. Some (e.g. *record*) handled by *syntax analysis*, others (e.g. *bass*) need wider semantics.
- Continuous speech pronunciation is affected by cross-word phonetic context. Handled by *phonological analysis*.

A multi-part pronunciation system is used. It consists of three components which are visited in order.

1. **Dictionary lookup** - limited size.
2. **Morphological rules** - split word into prefixes, roots and suffixes.
3. **Letter to sound rules** - essential for unknown words - highly complex, consider *hothouse*.

As we go down the list pronunciations may become less accurate, but typically will cover more words.

Intonation

This component is responsible for making the speech sound natural. It needs to control the perceived *pitch*, *stress* and *rhythm*. These are controlled by the physical quantities

1. **Duration** - normally determined by pre-stored durations, but may vary to improve rhythm.
2. **Fundamental frequency contour** - highly important. Normally constructed by choosing an "appropriate" phrase prototype then adding rise/falls to indicate stress.
3. **Energy contour** - important, but scaling proportionally to the fundamental frequency contour is commonly sufficient.

Stress is primarily taken from dictionary. Some systems allow additional stress to be applied to mark focus.

Current systems have limited capability for generating natural intonation and usually a very neutral ("boring") tone is adopted.

Phonological Analysis

This is the final symbolic processing stage. The primary function of this stage is to map phones into context specific versions called *allophones*.

Examples:

- A word final `d` sound will normally have a burst release. However, if the next word starts with a stop, then the burst is suppressed e.g. "red toy"
- A liquid `l` is *velarised* when it follows a vowel provided that there is not a stressed vowel immediately after e.g. ccompare the `l` in "pill" and "pillow".
- A word final `t` is usually *glottalised* when followed by a stressed vowel e.g. "that one".

Synthesis

The synthesis stage of a TTS system must:

1. convert the allophone symbol sequence into a continuous speech waveform;
2. apply the correct intonation.

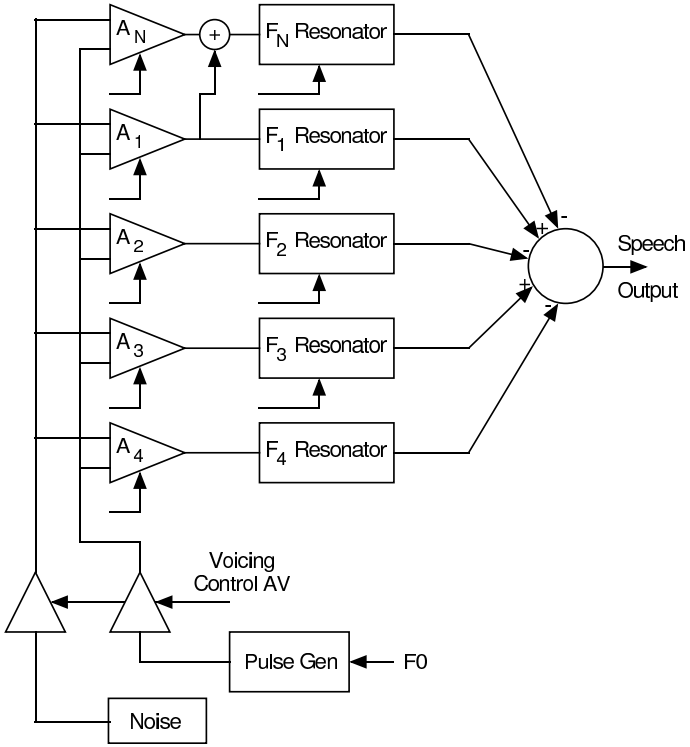
Common forms of synthesis are:

- **Formant-based**: both parallel and serial formant synthesisers. Vocal tract filter formed combining a number of resonances (formants).
- **Articulatory**: attempts to model the speech production system. Models of the articulators and vocal cords used - original "speech machine".
- **Concatenative**: explicitly use *synthesis units*. A representative example of each unit is stored and used for synthesis.

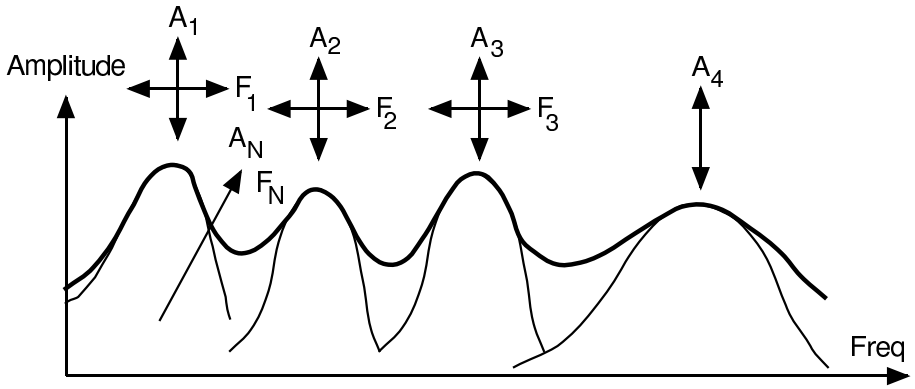
Many of these make use the *source-filter* model for speech production.

Formant-Based Synthesis

In formant-based synthesisers the symbol sequence is converted into a sequence of parameter vectors. Parallel formant synthesisers are preferred to serial synthesisers because they allow the formant amplitudes to be individually controlled.



A parallel formant synthesiser



The formant parameters

These parameters are updated every 10 msecs :

`F0 AV A1 F1 A2 F2 A3 F3 A4 AN FN`

- `F0` is fundamental frequency
- `AV` is the voicing level
- `Ai/Fi` is the amplitude and frequency of the `i`'th formant.
- `AN/FN` is a very broad low-frequency resonance which can be added to boost low frequencies (e.g. for nasals).
- `F4` is fixed in frequency and is included to improve naturalness.
- Adjacent formants are added in anti-phase. This is done to compensate for the 180° phase shift which occurs in the resonating channel.

The evolution of each parameter over time is computed from a set of tables, one for each allophone. Each table contains

- a target parameter set for the formant frequencies;
- transition information from the current allophone to the next to be computed.

Formant-based synthesis:

- Computationally very efficient, low cost implementations are possible;
- Naturalness of the output is limited, but has been widely integrated in TTS.
- Large number of coupled parameters to tune. Formant frequencies and bandwidths are inherently difficult to estimate from speech data.

Concatenative Synthesis

As memory costs have dropped, it has become feasible to store many more sounds in memory and this has led to concatenative synthesis. Here, "blocks" of "speech data" are joined together to form the complete sentence.

Two important questions:

1. **Synthesis units**: the choice of synthesis units is important. Trade off between longer and shorter units. Multiple examples of the same synthesis unit may be stored and selected, for example, according to duration and pitch contour.
2. **Representation of units**: how is each of the units to be stored. Common forms are linear prediction coefficients and waveforms.

A wide range of concatenative synthesis schemes are been built. Many commercially available systems are based on this approach.

Synthesis Units

The choice of synthesis units balances number against length

- long units have few concatenation points, typically also have smaller concatenation discontinuities.
- long units have longer time-scale "naturalness".
- fewer short units are required.

Units considered are similar to those for speech recognition (discussed later). One popular unit in synthesis is the **diphone**.

A diphone is a segment starting at the centre of one phone and ending at the centre of the next.

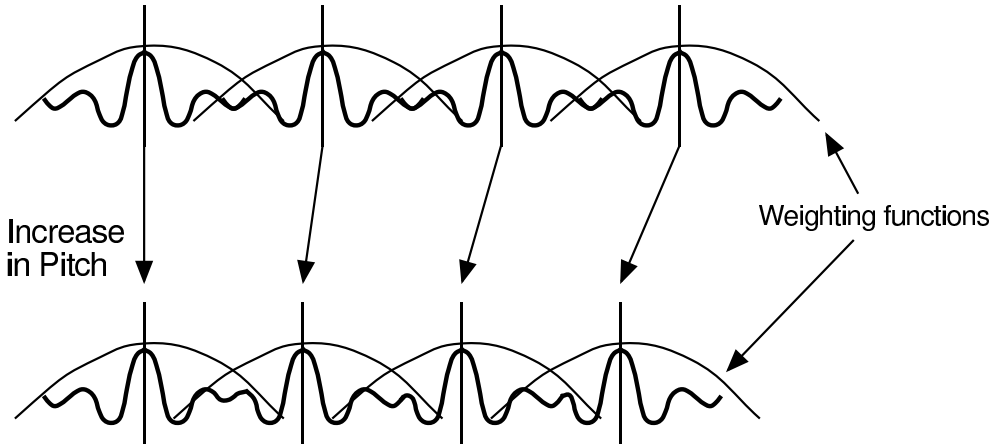
Diphones units are desirable as they:

1. preserve transitions between phones - otherwise difficult to synthesise;
2. have concatenation points in the middle of phones, which are spectrally stable. This tends to result in only small concatenation discontinuities.
3. may be simply derived from real speech - more natural output should be possible.

Waveform Concatenation

In waveform concatenation speech is not synthesised, pre-recorded segments are smoothly concatenated while enabling the correct intonation to be applied. Pitch Synchronous Overlap and Add is a common waveform concatenation scheme. This section describes **Time-Domain PSOLA**. Compared to LP synthesis it:

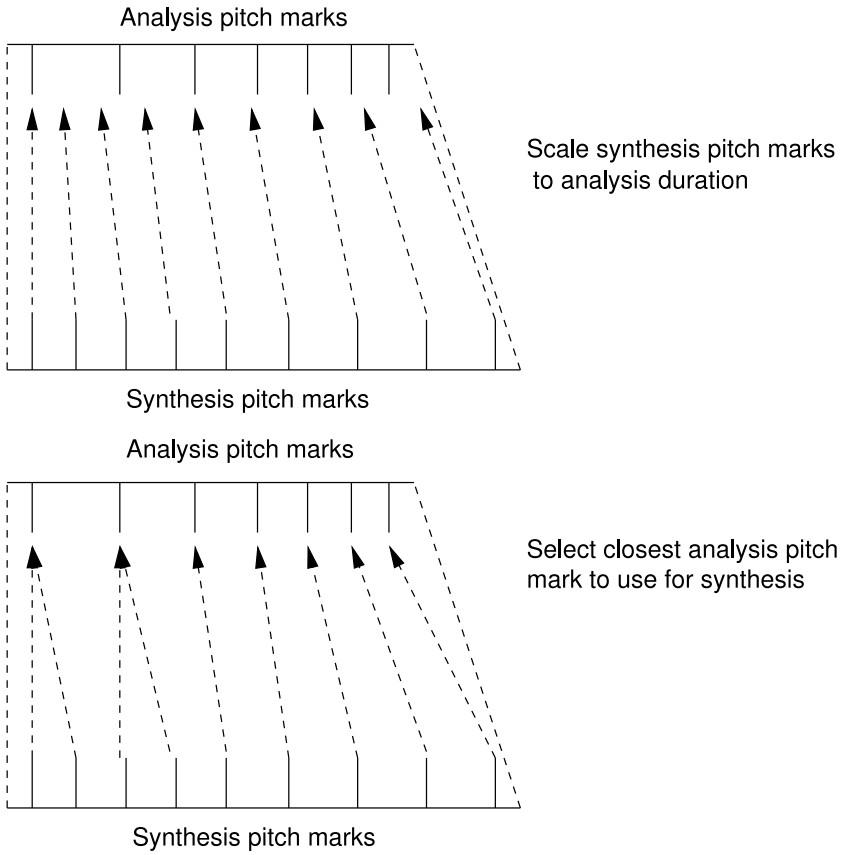
- requires large amounts of memory to store waveform;
- does not allow spectral smoothing at concatenation points - choice of synthesis unit is very important;
- does not rely on the source-filter model approximation;



Pitch Synchronous Overlap and Add (PSOLA)

The basic procedure is:

1. break the natural speech signal up by applying Hanning windows pitch synchronously in voiced regions and evenly spaced in unvoiced regions;
2. raise or lower pitch and alter duration according to the required intonation (figure 3);
3. add the sequences together compensating for the number and amplitude contributing to the synthetic signal.



Altering Pitch and Duration in PSOLA

HMM-based Text-to-Speech Synthesis

HMMs have been described so far in terms of their use in recognition :

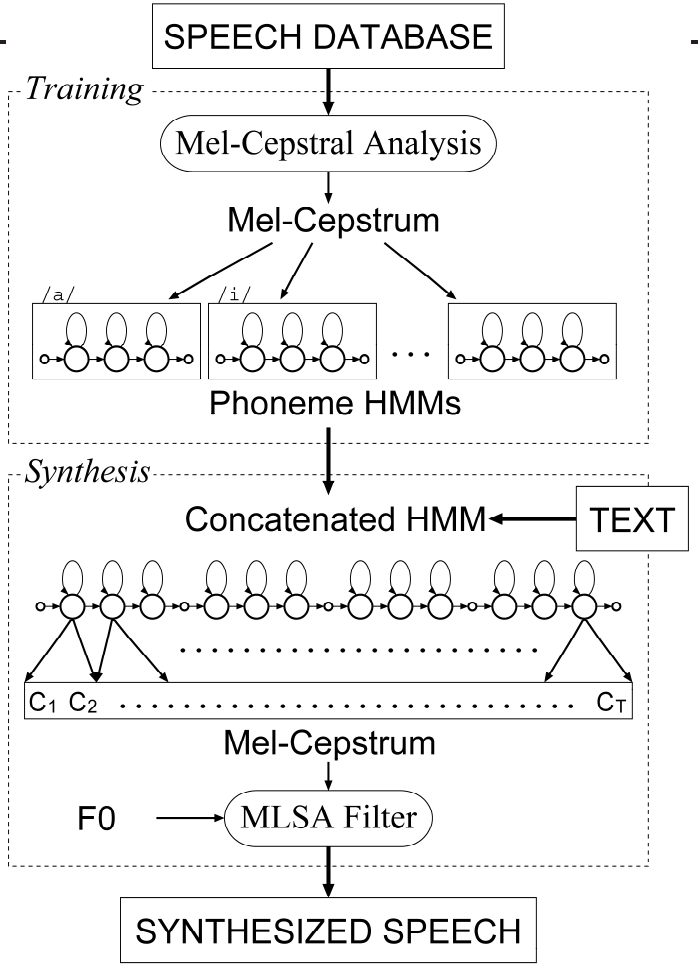
- For an acoustic sequence O_1^T , find the most likely word sequence W as

$$\underset{W}{\text{argmax}}\ P(W|O_1^T) = \underset{W}{\text{argmax}}\ P_{\text{HMM}}(O_1^T|W)\ P_{\text{LM}}(W)$$

Text-to-Speech Synthesis can also be formulated statistically:

- For a word sequence W , find the most likely acoustic sequence $\widehat{O_1^T}$ as

$$\underset{O_1^T}{\text{argmax}}\ P(O_1^T|W) = \underset{O_1^T}{\text{argmax}}\ P_{\text{HMM}}(O_1^T|W)$$



HMM-based TTS Architecture

- Words are mapped to phonemes
- 'Concatenated' ('composite') HMM determined by the phoneme sequence
- MLSA – Mel Log Spectrum Approximation

Generation of Acoustic Sequences

For a word sequence W, find a good state sequence through the composite model:

$$\widehat{Q} = [s_1\, s_1\, s_1 \underbrace{s_2\, s_2\, s_2\, s_2}_{d_2=4}\, s_3\, s_3 \ldots s_N\, s_N\, s_N]$$

- State durations are determined by models for phone/state duration

$$P(Q|\lambda) = \prod_{i=1}^{N} p_i(d_i)$$

- Cepstral sequences are generated given the HMM state sequence $\widehat{Q}$

$$\widehat{O_1^T} = \underset{O_1^T}{\operatorname{argmax}}\, P(O_1^T|\widehat{Q})$$

HMM Independence Assumptions Can Produce Discontinuous Feature Sequences

Recall that $P_{\text{HMM}}(O_1^T|Q_1^T) = \prod_{t=1}^{T} P(O_t|Q_t)$.

- For Gaussian observation distributions, the most likely observation at each time is simply the mean of the Gaussian distribution for that state

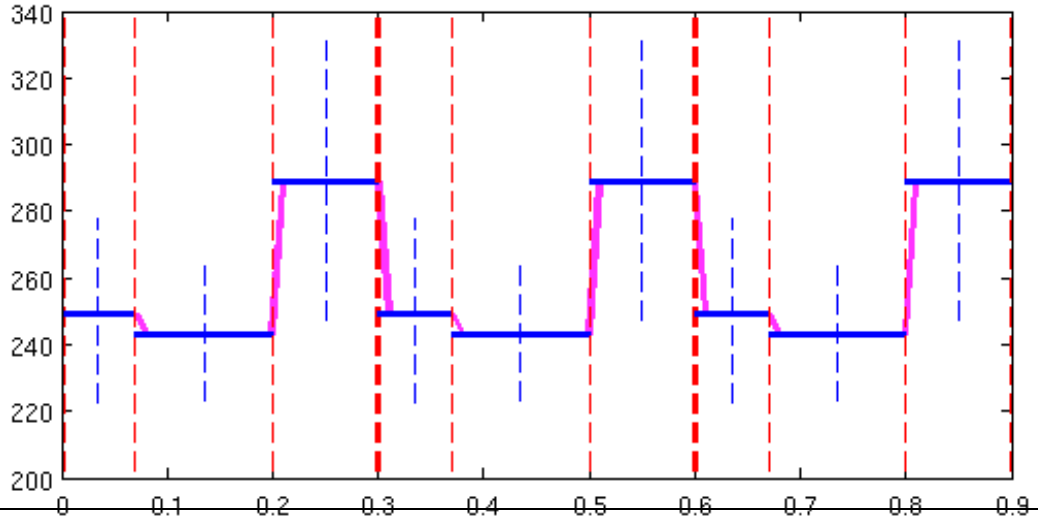
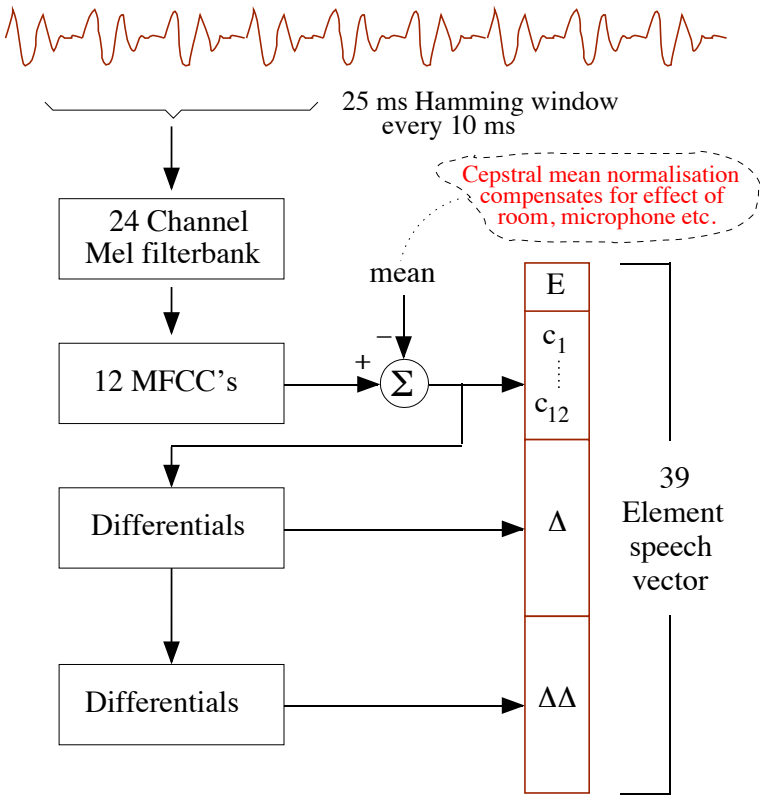


Figure 15.13 Synthesis of an F0 contour from an HMM, where we ignore the dynamic information. The thick vertical lines represent model boundaries and the thin lines state boundaries. The horizontal lines indicate the state means. It is clear that in all cases the algorithm generates the state means, which results in discontinuities at state boundaries.

From Taylor, Chapter 15

Solution: Generate Dynamic Features

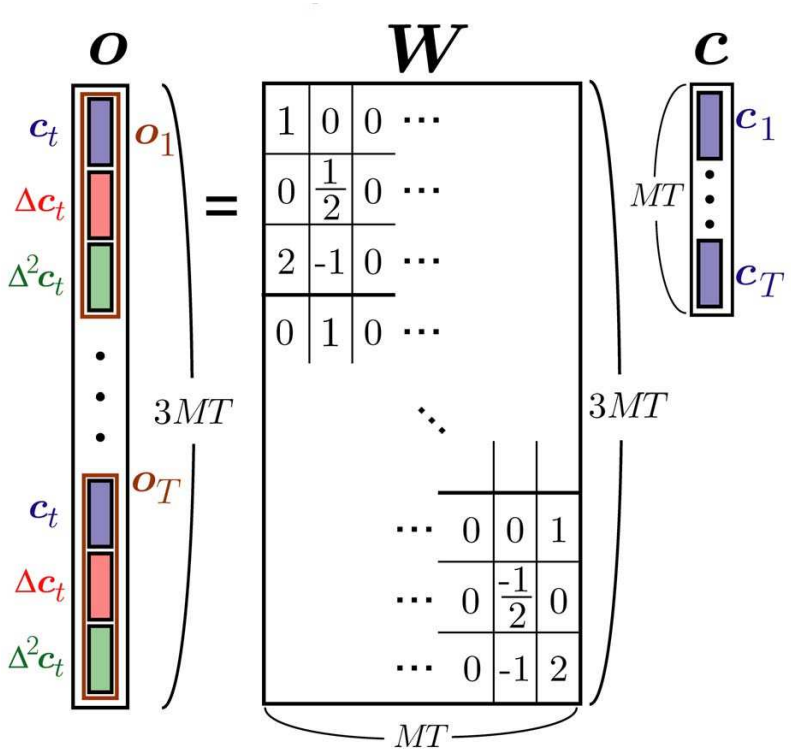
Recall:



Dynamic Features (Differentials) are Found from Static MFCCs

Optimising Over Consistent Dynamic Features[1]

Find $\widehat{O_1^T} = W\,\widehat{c_1^T}$ where $\widehat{c_1^T} = \text{argmax}_{c_1^T} P(W\,c_1^T \,|\, \widehat{Q}_1^T)$



Dynamic Features (Differentials) are Found from Static MFCCs

[1] http://www.sp.nitech.ac.jp/ tokuda/tokuda_iscslp2006.pdf

Enforcing Consistency in Cepstral Sequences and their Delta Coefficients Leads to Smoother Trajectories

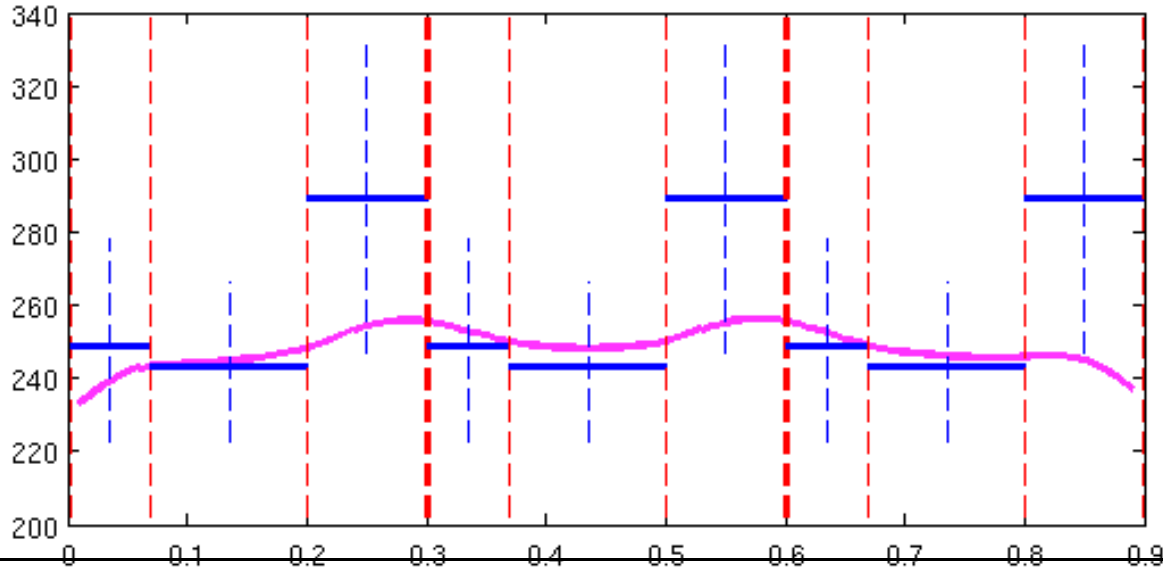


Figure 15.14 Synthesis of an F0 contour where we use Equation 15.31 and model the dynamic behaviour. The contour is now clearly continuous and the state boundaries are not observable from the contour alone. Only in some cases is the mean value for a state reached.

From Taylor, Chapter 15 - see that chapter for solution details

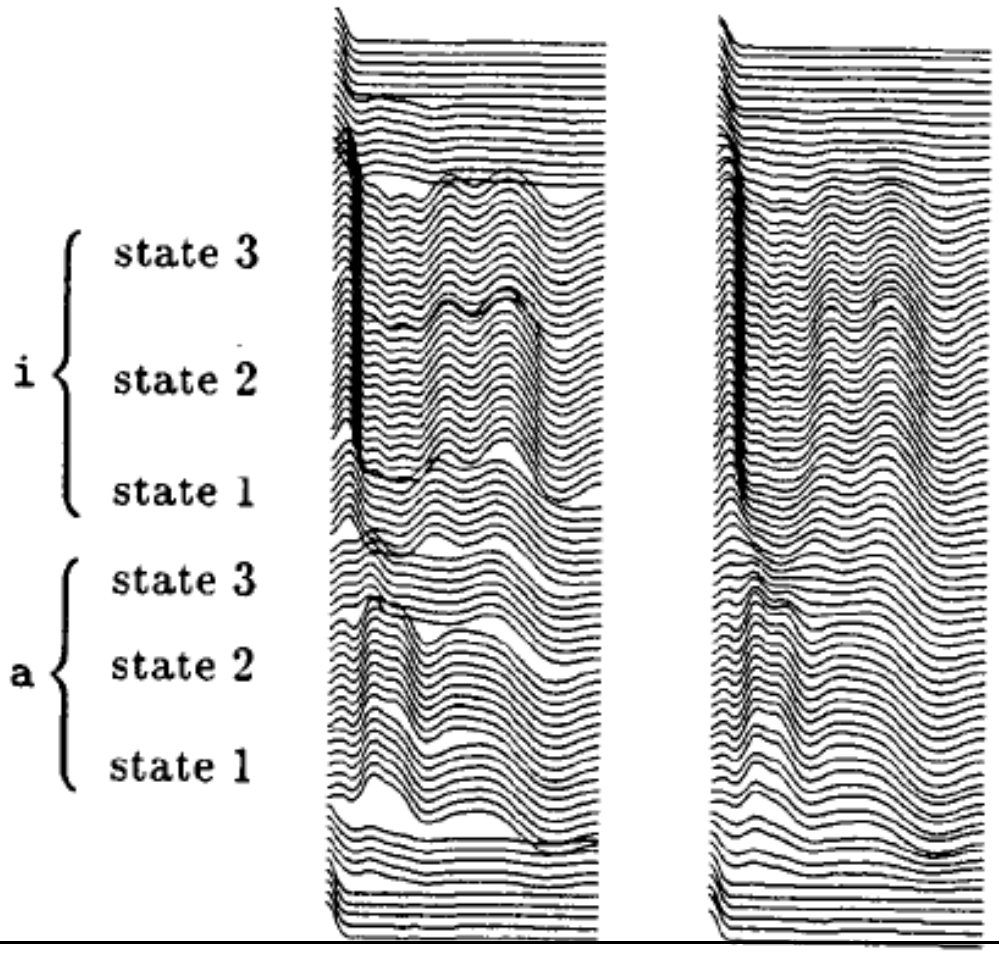


Figure 15.16 Comparison of evolution of generated spectra between synthesising from the mean (left figure) and when using dynamic information (right figure). In the left figure, we can clearly see that the shape of the spectra jumps at state boundaries. Once we include dynamic information the evolution is smooth across the utterance.

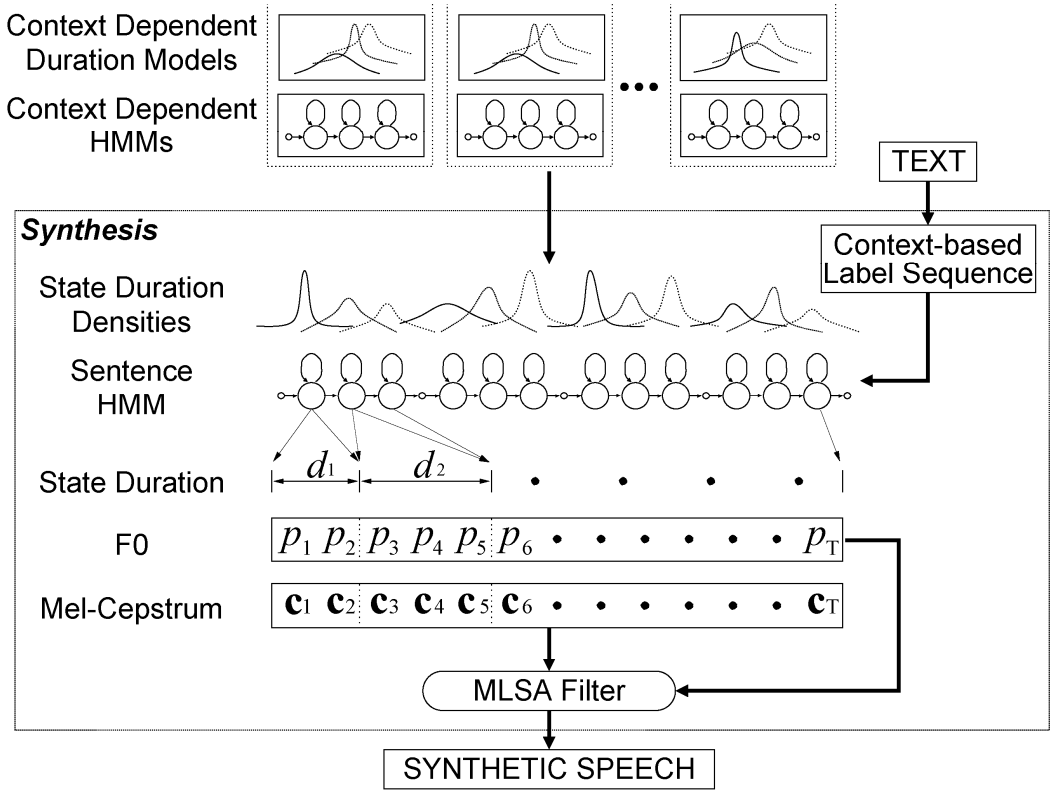
From Taylor, Chapter 15

Context Clustering Questions Have Longer Span and Richer Linguistic Knowledge[2]

- {preceding, current, succeeding} phoneme
- Position of current phoneme in current syllable
- Number of phonemes at {preceding, current, succeeding} syllable
- Accent of {preceding, current, succeeding} syllable
- Position of current syllable in current word
- Number of {preceding, succeeding} stressed syllables in current phrase
- Number of {preceding, succeeding} accented syllables in current phrase
- Number of syllables {from previous, to next} stressed syllable
- Number of syllables {from previous, to next} accented syllable
- Vowel within current syllable
- Guess at part of speech of {preceding, current, succeeding} word
- Number of syllables in {preceding, current, succeeding} word
- Position of current word in current phrase
- Number of {preceding, succeeding} content words in current phrase
- Number of words {from previous, to next} content word
- Number of syllables in {preceding, current, succeeding} phrase
- Position in major phrase
- ToBI endtone of current phrase

[2] http://www.sp.nitech.ac.jp/ tokuda/tokuda_iscslp2006.pdf

How HMMs in TTS Differ from ASR



In TTS: F_0 is a separate stream; explicit state duration densities' MLSA Filter; single mixture Gaussian observation densities; HMM state sequence Q is generated from words via Linguistic Analysis (see earlier)

Advantages of HMM-based TTS

- Data-driven model estimation techniques replace hand-crafting of rules
- Fast generation of synthesis systems from data collections
- Models can be adapted
  - rapid training/enrollment for new speakers
  - speaker interpolation (voice mixing and voice morphing)
  - multilingual and cross-lingual speech synthesis
- Shared modelling strategies for ASR and Synthesis

W.J. Byrne
P.C. Woodland
M.J.F Gales
T. Hain