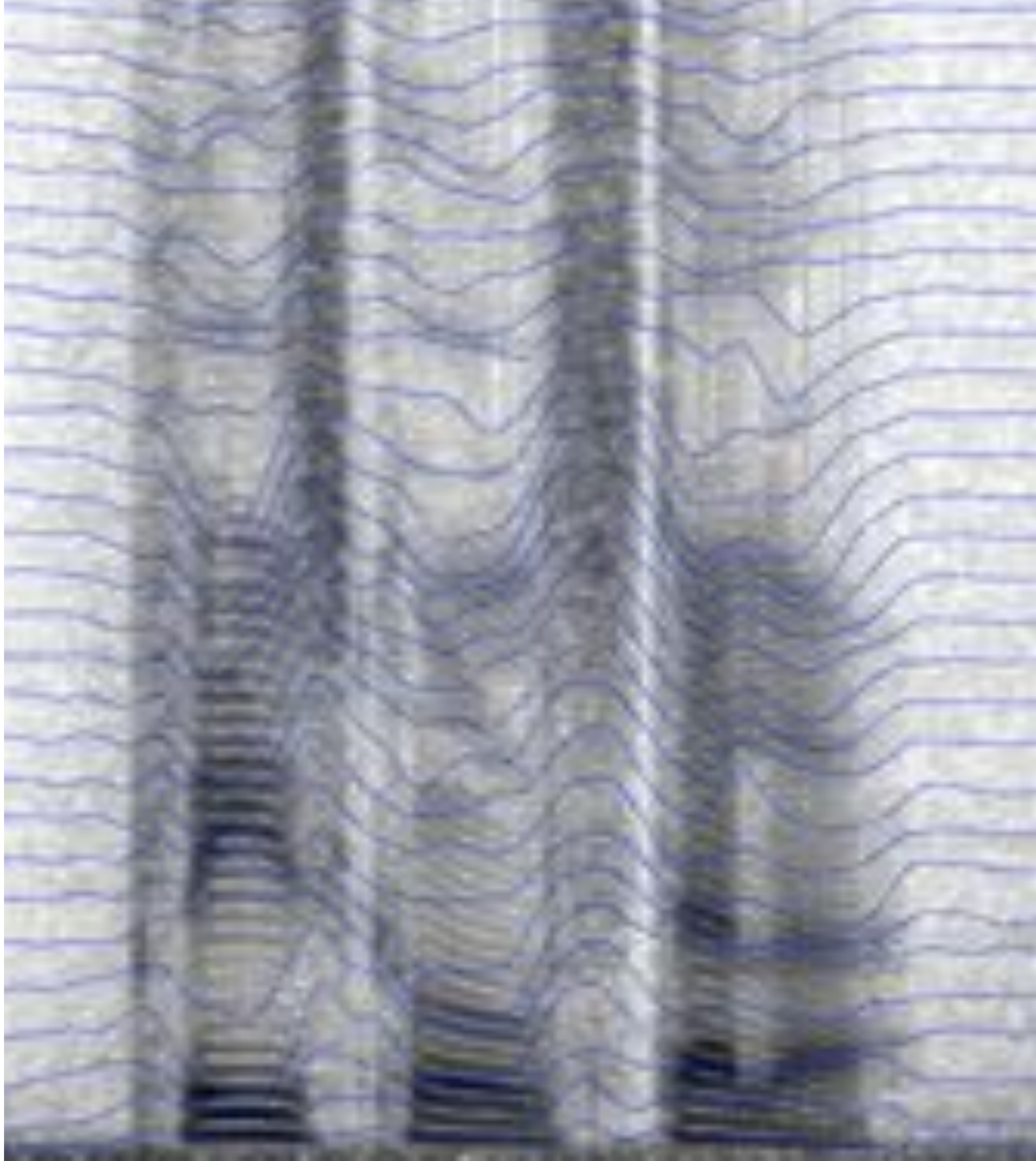
Speech Synthesis

Simon King
University of Edinburgh



Databases for speech synthesis

- key concepts
- script design
- annotating the database

What you should already know

- the front end
  - linguistic specification
- unit selection method
  - select units from similar contexts
  - target cost measures this similarity
- basic Automatic Speech Recognition
  - Hidden Markov Models
  - finite state language model
  - decoding

Databases for speech synthesis

- key concepts
- script design
- annotating the database

Key concept: base unit type

- **relatively small number of types**
  - e.g., diphone

- in unit selection
  - base unit type is **strictly matched** between target and candidate
  - unless database is badly designed: then we would have to *back off* to a similar type

- therefore, target cost does **not** need to query the base unit type
  - only query its context

Key concept: context

- the linguistic and acoustic environment in which a base unit occurs, including
  - **phonetic context** - the sounds before and after it
  - **prosodic environment** - stress, prosody, …
  - **position** - in the syllable, word, phrase, …

- Exact features considered will depend on target cost formulation (e.g., IFF or ASF)

Key concept: coverage

- We would like a database of speech which contains
  - every possible speech base unit type
  - in every possible context

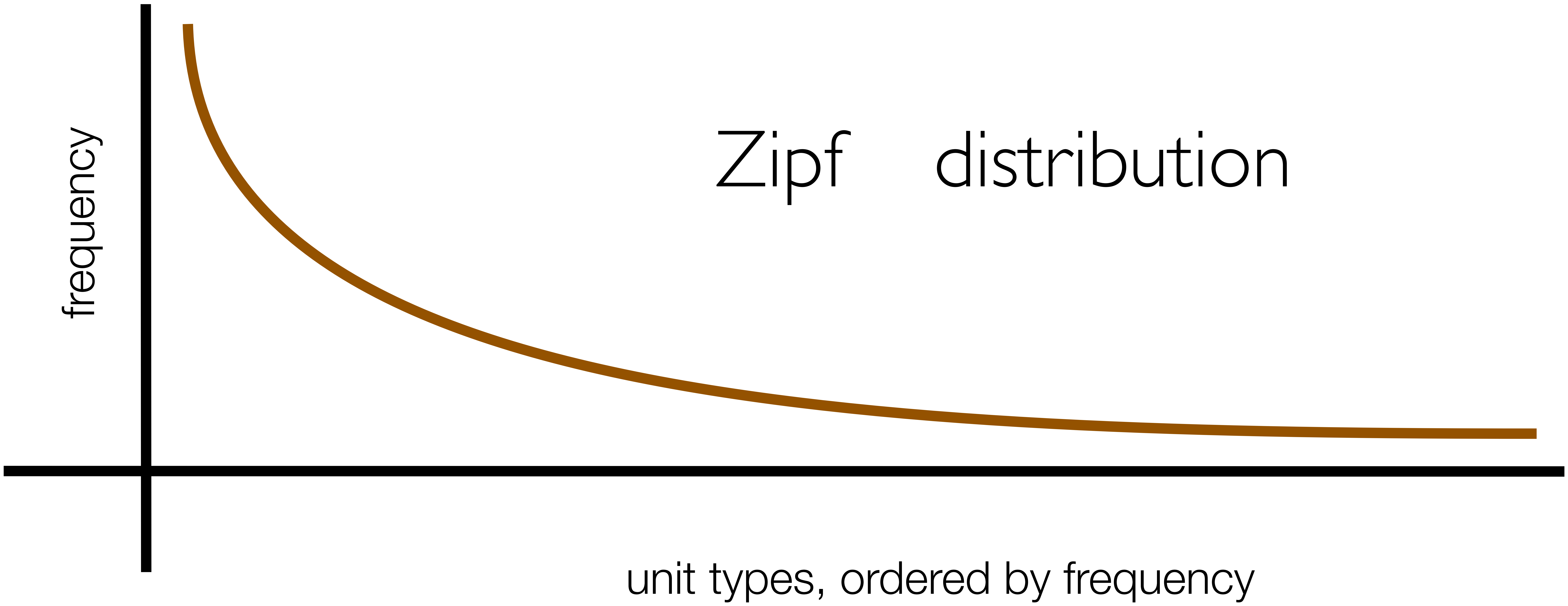
- This list of desired *base unit types in context* will be **very, very long**
  - if we limit the scope of context, the list will be finite

- Will it be possible to record one example of every *unit-in-context*?

Large number of rare events

- A few types (units-in-context) are very frequent
- A large number of types are individually very infrequent
  - but the *large number* of such types means that together they make up a significant proportion of (spoken) language

- There is a high chance that we will need **at least one** rare type in **any sentence** we have to synthesise

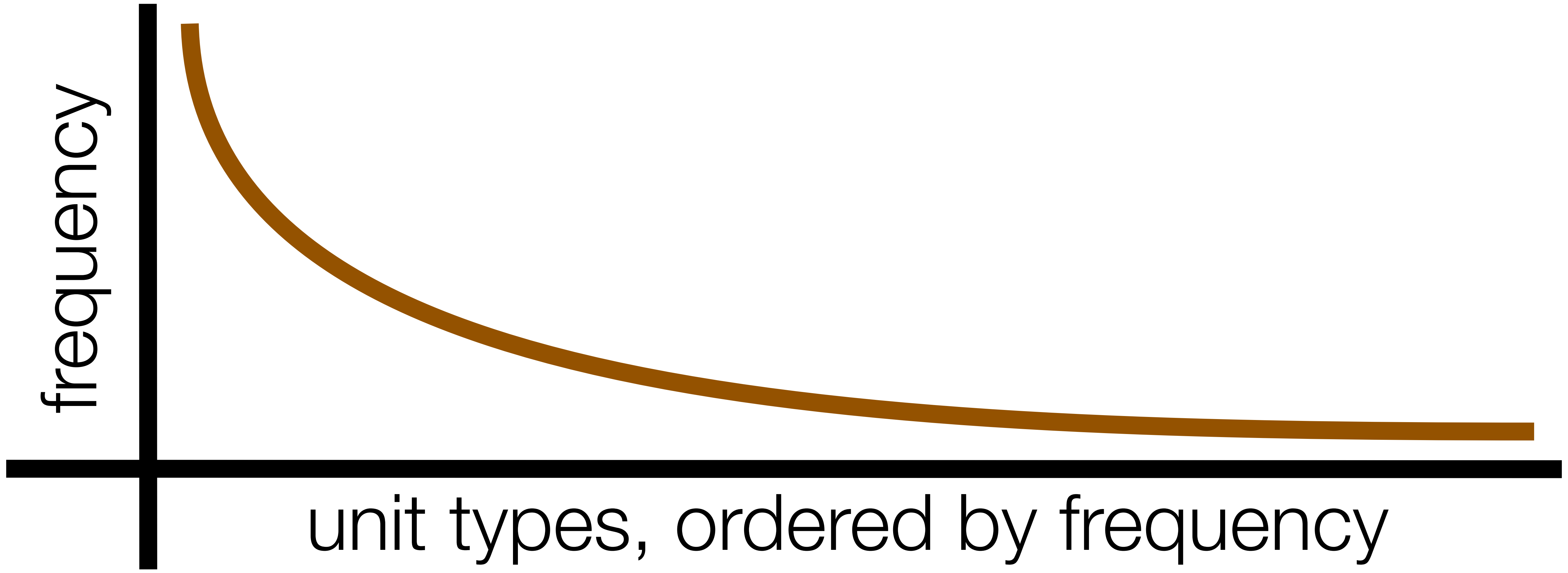


Zipf distribution
frequency
unit types, ordered by frequency

Databases for speech synthesis

- key concepts
- script design
- annotating the database

Why *design* a script?



- In randomly-chosen natural text
  - Zipf-like distribution of units-in-context
- As database size increases
  - number of **tokens** of frequent types increases **rapidly**
  - number of infrequent **types** with at least one example grows very **slowly**
  - many (most!) types will have **no tokens** at all, even for very large database sizes

- In practice, it will be impossible to find a set of sentences that includes at least one token of every unit-in-context type
- So, try to **design** a script that is better than random selection of sentences

Goals of script design

- Cover as **many types** (in context) as possible
  - increase chance of finding an exact match at synthesis time - *although still very unlikely*
  - maximise the variety of contexts in which each base unit type occurs
    - the target cost will differentiate between them
    - join cost has better chance of finding unit sequences that concatenate well

- With as **few tokens** as possible - i.e., in as few sentences as possible
  - recording speech is time consuming
  - harder to maintain consistency over longer recording periods (days, weeks, months)
  - in unit selection, the run-time system will include a copy of the database

Typical approach to script design: a greedy algorithm for text selection

1. Find a **very large text corpus**
    - e.g., newspaper text, out-of-copyright novels, web scraping
2. Make an exhaustive 'wish list' of all possible **types** (in context) that we would like
3. Find the sentence in the corpus which provides the largest number of different **types** that we don't already have
4. Add that sentence our recording script
5. Remove those types from the 'wish list'
6. If recording script is long enough, stop. Otherwise, go to 3.

Where do we get this “very large text corpus” ?

- Out of copyright literature (old novels)
  - e.g., as used in the ARCTIC corpora
- Newspaper text
  - usually copyrighted, so must obtain permission to use
- Problems with most sources of text:
  - written text is not usually intended to be read aloud
  - prosodic variation will therefore be limited
  - long sentences lead to insufficient phrase initial/final segments

Example of text selection

- We'll assume that we have a large corpus of text to start from
- **Corpus cleaning**
  - Define the vocabulary (e.g., only words in our dictionary, or the most frequent words in the corpus)
  - Discard all sentences that contain out-of-vocabulary (OOV) words
  - Discard all sentences that are too long (*hard to read out loud*) or too short (*atypical prosody*)
  - *Optional: discard hard-to-read sentences*

- **Front-end processing**
  - Pass the text through the TTS front end to obtain, for each sentence
    - base unit sequence (e.g., diphones)
    - linguistic context of each unit (e.g., stress)

The wish list

- Define the base unit **type** - let's use diphones in this example
- Which would give this wish list:

```
aa_aa  aa_f               zh_f    zh_p
aa_ae  aa_g               zh_g    zh_r
aa_ah  aa_hh              zh_hh   zh_s
aa_ao  aa_ih              zh_ih   zh_sh
aa_aw  aa_iy              zh_iy   zh_t
aa_ay  aa_jh   •••   ••   zh_jh   zh_th
aa_b   aa_k               zh_k    zh_uh
aa_ch  aa_l               zh_l    zh_uw
aa_d   aa_m               zh_m    zh_v
aa_dh  aa_n               zh_n    zh_w
aa_eh  aa_ng              zh_ng   zh_y
aa_er  aa_ow              zh_ow   zh_z
aa_ey  aa_oy              zh_oy   zh_zh
```

The wish list

- In reality we want every type in every context
- What context? - let's just consider stress
- Which would give this wish list:

```
aa_aa_unstressed    aa_b_stressed
aa_aa_stressed      aa_ch_unstressed
aa_ae_unstressed    aa_ch_stressed
aa_ae_stressed      aa_d_unstressed
aa_ah_unstressed    aa_d_stressed
aa_ah_stressed      aa_dh_unstressed
aa_ao_unstressed    aa_dh_stressed
aa_ao_stressed      aa_eh_unstressed
aa_aw_unstressed    aa_eh_stressed
aa_aw_stressed      aa_er_unstressed
aa_ay_unstressed    aa_er_stressed
aa_ay_stressed      aa_ey_unstressed
aa_b_unstressed     aa_ey_stressed
```

●●● etc.

Create an index of all available sentences, and the units they contain

So I came here.	sil_s s_ow ow_ay ay_k k_ey ey_m m_hh hh_ih ih_r r_sil
Now we have finally heard her.	sil_n n_aw aw_w w_iy iy_hh hh_ae ae_v v_f f_ay ay_n n_ax ax_l l_iy iy_hh hh_er er_d d_hh hh_er er_sil
Those chefs know who they are.	sil_dh dh_ow ow_z z_sh sh_eh eh_f f_s s_n n_ow ow_hh hh_uw uw_dh dh_ey ey_aa aa_r r_sil
…etc	

Select richest sentence. Move it to the script. Update wish list

So I came here.	sil_s s_ow ow_ay ay_k k_ey ey_m m_hh hh_ih ih_r r_sil
Now we have finally heard her.	sil_n n_aw aw_w w_iy iy_hh hh_ae ae_v v_f f_ay ay_n n_ax ax_l l_iy ~~iy_hh~~ hh_er er_d d_hh ~~hh_er~~ er_sil
Those chefs know who they are.	sil_dh dh_ow ow_z z_sh sh_eh eh_f f_s s_n n_ow ow_hh hh_uw uw_dh dh_ey ey_aa aa_r r_sil
…etc	

```
aa_aa    aa_f          ay_ey          ey_f          hh_f          zh_f    zh_p
aa_ae    aa_g          ay_f           ey_g          hh_g          zh_g    zh_r
aa_ah    aa_hh         ay_g           ey_hh         hh_hh         zh_hh   zh_s
aa_ao    aa_ih         ay_hh          ey_ih         hh_ih         zh_ih   zh_sh
aa_aw    aa_iy         ay_ih          ey_iy         hh_iy         zh_iy   zh_t
aa_ay    aa_jh   •••   ay_iy    •••   ey_jh   •••   hh_jh   •••   zh_jh   zh_th
aa_b     aa_k          ay_jh          ey_k          hh_k          zh_k    zh_uh
aa_ch    aa_l          ay_k           ey_l          hh_l          zh_l    zh_uw
aa_d     aa_m          ay_l           ey_m          hh_m          zh_m    zh_v
aa_dh    aa_n          ay_m           ey_n          hh_n          zh_n    zh_w
aa_eh    aa_ng         ay_n           ey_ng         hh_ng         zh_ng   zh_y
aa_er    aa_ow         ay_ng          ey_ow         hh_ow         zh_ow   zh_z
aa_ey    aa_oy         ay_ow          ey_oy         hh_oy         zh_oy   zh_zh
```

Optional improvements

- Guarantee **at least one token** of every base unit type

- Try to cover the **rarest** units first
  - more common units will be selected anyway, as a by-product
- How to define "rarest"?
  - count occurrences in the original large corpus
- How to implement this
  - include weights in the "richness" measure that reward rarer units in inverse proportion to their frequency

Optional: domain-specific script

1. Select (or manually design, or automatically generate) in-domain sentences
2. Measure coverage obtained so far
3. Fill in the gaps in coverage, using sentences selected from the large text corpus

Databases for speech synthesis

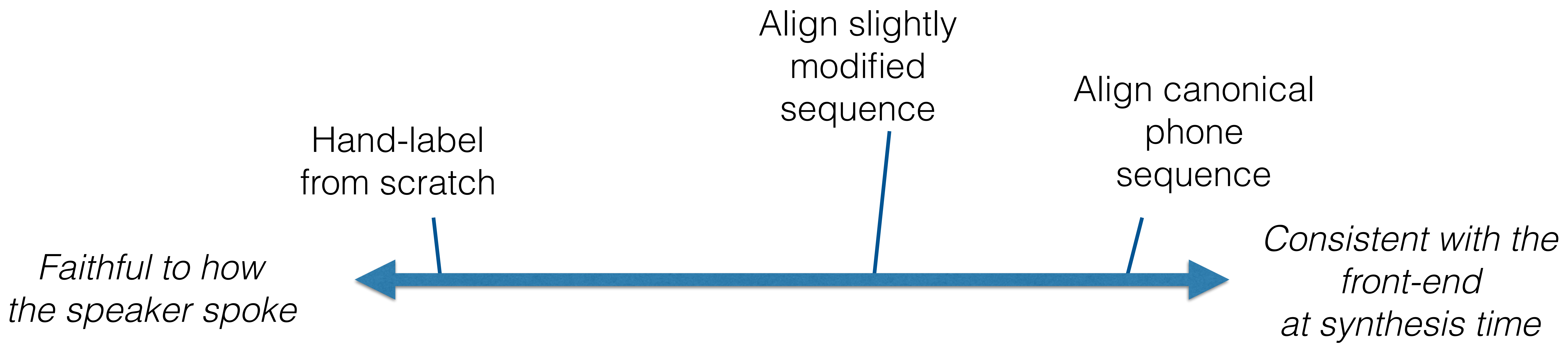
- key concepts
- script design
- annotating the database

Orientation

- What have we got?
  - a script composed of sentences
  - a recording of each sentence

- What remains to be done?
  - a time-aligned phonetic transcription of the speech
  - annotate the speech with supra-segmental linguistic information

Why not simply hand-label the speech ?



"The text sentence that the speaker read out"

Analytical labelling

- Two reasons we prefer analytical labelling to be done automatically
  - more <u>consistent</u>
    - the labels are **just an index** to retrieve units from the database
    - we want the **predictions from text** to match the labels on the database
  - faster and <u>cheaper</u>

- manual correction of automatic labels
  - standard practice in some commercial systems
  - mainly this involves fixing gross errors such as mis-alignment

Forced alignment

- Standard technique from Automatic Speech Recognition
- The same as full-blown speech recognition, except
  - we have a very highly constrained language model
    - because we know the word sequence
  - during decoding, we record the model- (or state-) level alignment

Ingredients for forced alignment

- <u>Acoustic models</u>
  - a fully-trained set of phone models
- <u>Pronunciation model</u>
  - the same dictionary we will use for synthesis
  - can include pronunciation variation
  - plus optional rule-based variations, such as vowel reduction
- <u>Language model</u>
  - constructed from the known word sequence for the current sentence
    - i.e., language model is different for each sentence
    - can insert optional silences between words

Language model

`there      was      a      change      now`

Pronunciation model = dictionary + optional vowel reduction

"…what **can** it do for…"

k ae n

ax

Acoustic model

- Could borrow from an existing ASR system
  - actually tend to get better results with simpler, **speaker-dependent models**
  - trained on the speech database

- Hang on: training on the "test data"? Isn't that cheating?
  - no - because it's not "test data" !

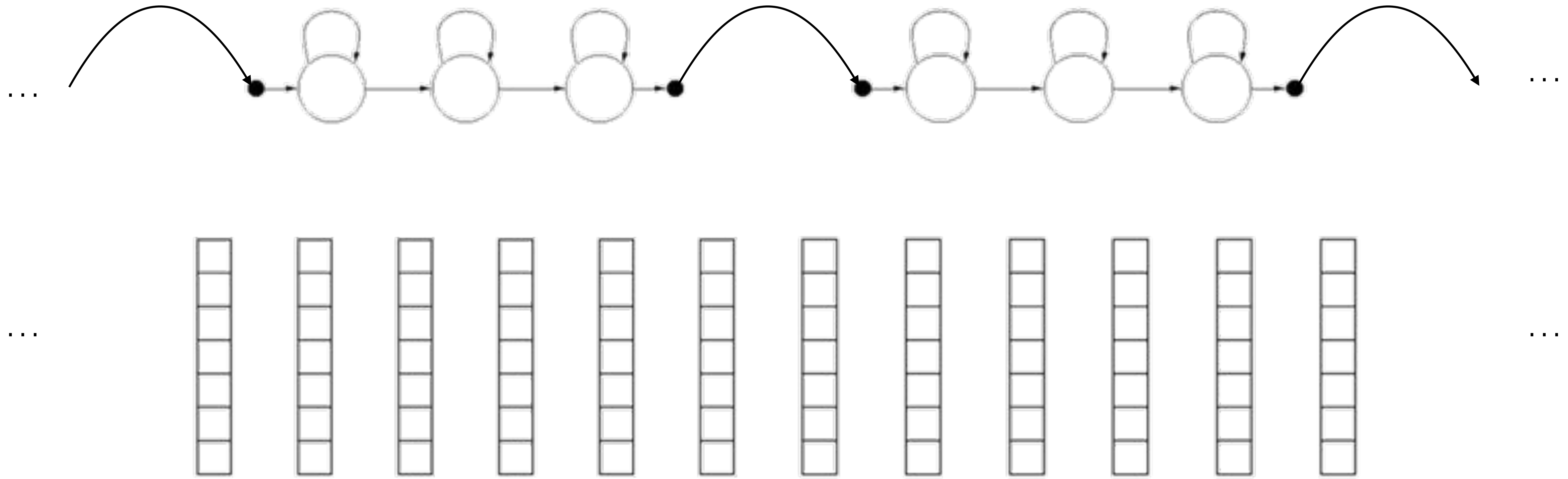
Training an acoustic model on the recorded speech data

- We only have word transcriptions
- Aligned at sentence boundaries

- We already know about basic Automatic Speech Recognition
  - training models on data with model-level (e.g., whole word) aligned transcriptions
- But, even there, we *did not need state-level alignments*

- Can generalise this to not needing model-level alignments
  - concatenate models, to make an acoustic model for a particular whole **sentence**
  - this is just a single (albeit rather long) HMM, and we know how to train that

Flat start training of HMMs



Optional silences between words

An acoustic model of optional silences, to use between words

- Single-state model
- Emission parameters (e.g., a GMM) are tied to the centre state of the long silence model
- Skip transition allows model to emit a sequence of **zero or more** observations

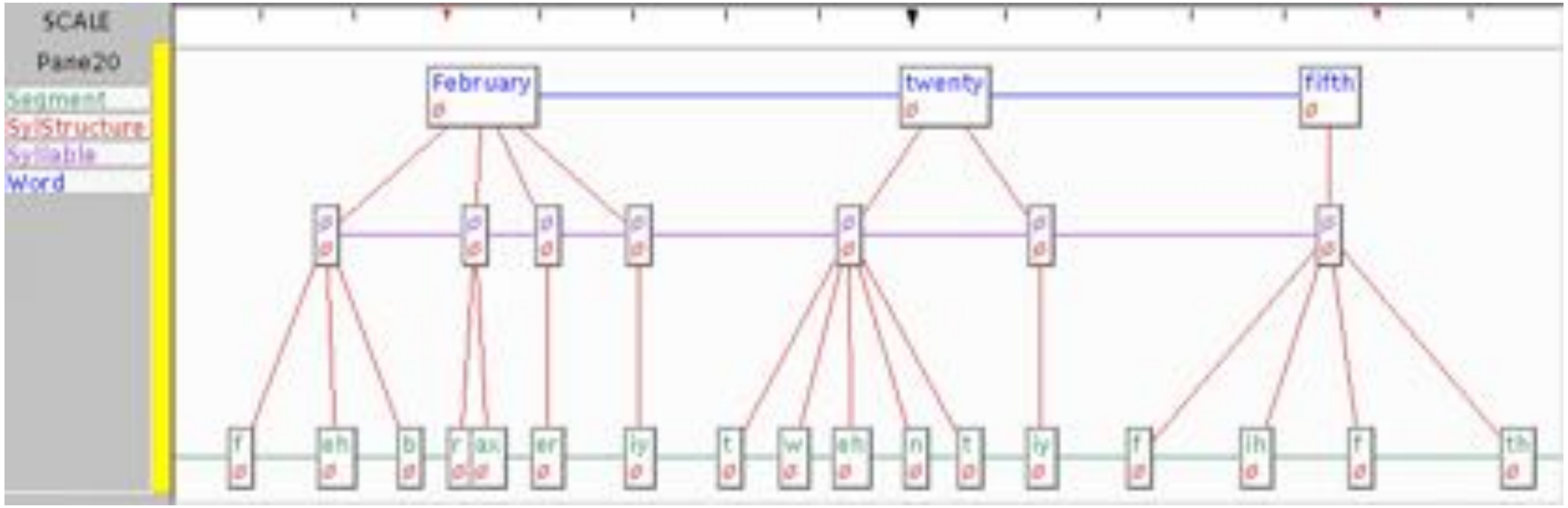
Compile language model + pronunciation model + acoustic model

```
        there      was      a      change      now

sil  dh eh r    w aa z    ax    ch ey n jh    n aw   sil

        ax    sp    ax    sp    sp    ax          sp    ax
```

Combining aligned phone sequence with supra-segmental structure



sil f eh b r ax er iy sp t w eh n t iy f ih f th sil

What next?

- How good is our synthetic voice?

- It's time to evaluate it, but how?
  - listen to it ourselves?
  - ask others to listen to it?
  - measure objectively?

- What precisely do we want to measure? Why?

What next?

- How good is our synthetic voice?

- It's time to evaluate it, but how?
  - listen to it ourselves?
  - ask others to listen to it?
  - measure objectively?

- What precisely do we want to measure? Why?

Can we judge that in isolation, or must it be in comparison to another system/voice?

What next?

- How good is our synthetic voice?

- It's time to evaluate it, but how?
  - listen to it ourselves?
  - ask others to listen to it?
  - measure objectively?

Each of these has advantages and disadvantages that we need to consider.

- What precisely do we want to measure? Why?

What next?

- How good is our synthetic voice?

- It's time to evaluate it, but how?
  - listen to it ourselves?
  - ask others to listen to it?
  - measure objectively?

- What precisely do we want to measure? Why?

Naturalness?
Intelligibility?
Something else?