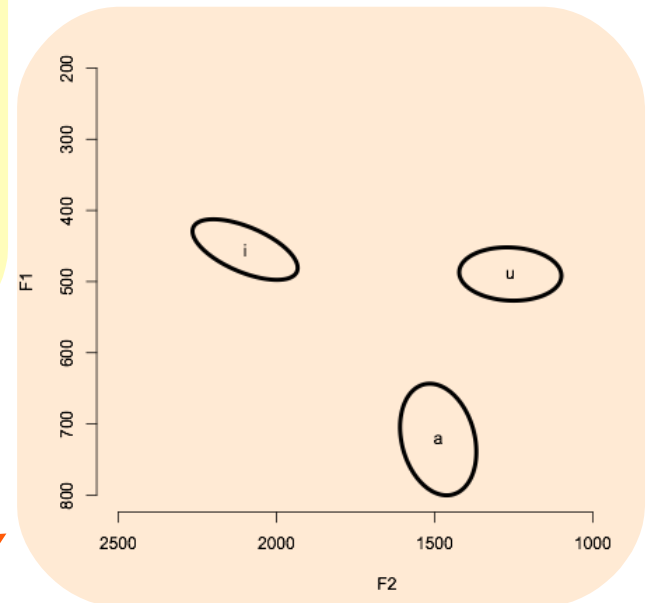
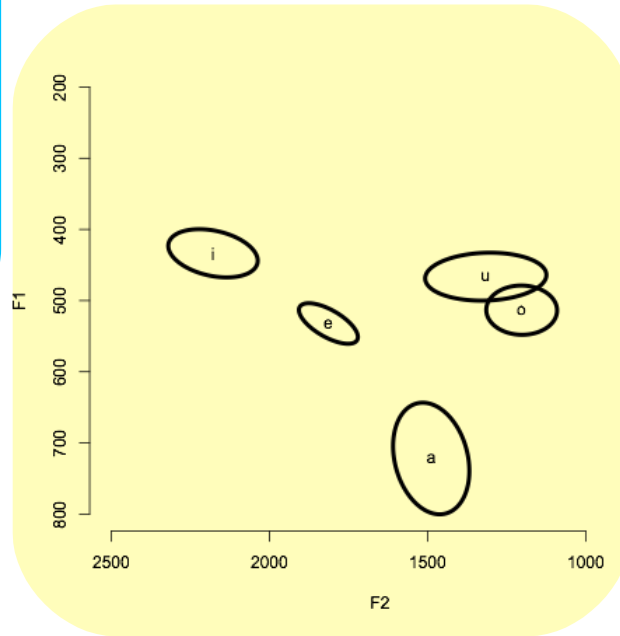
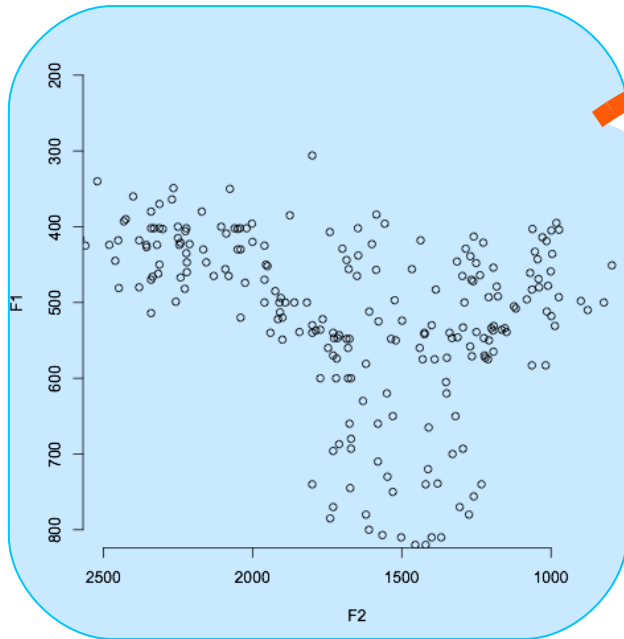
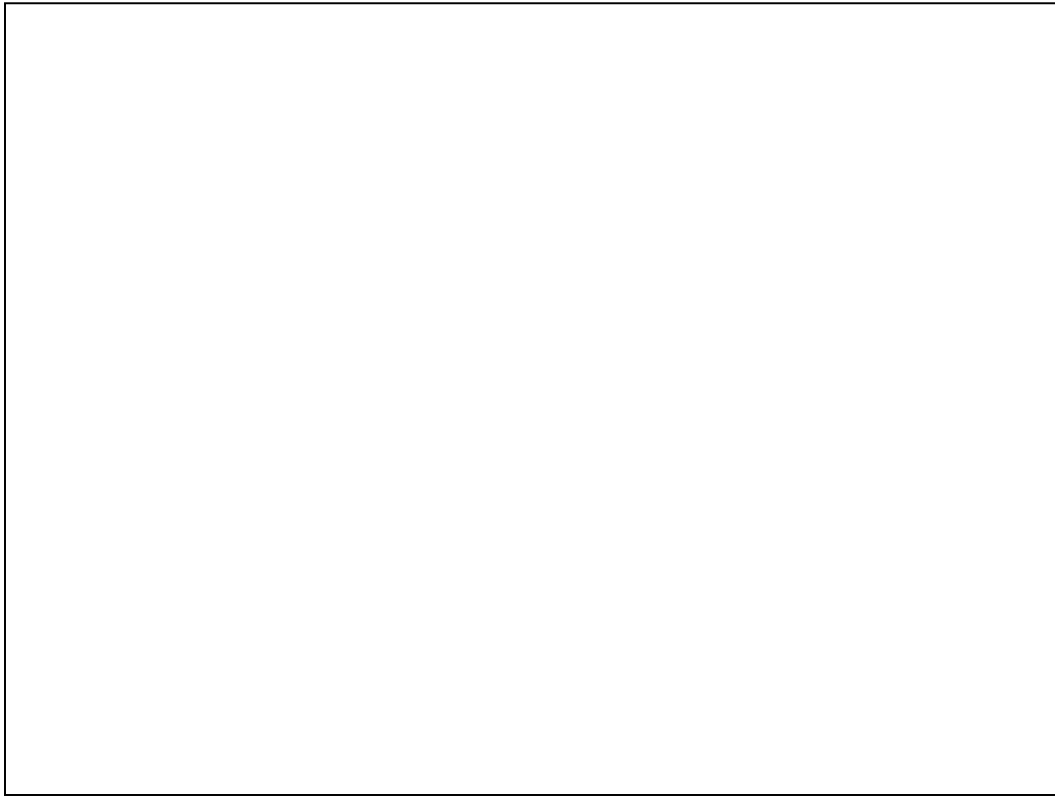


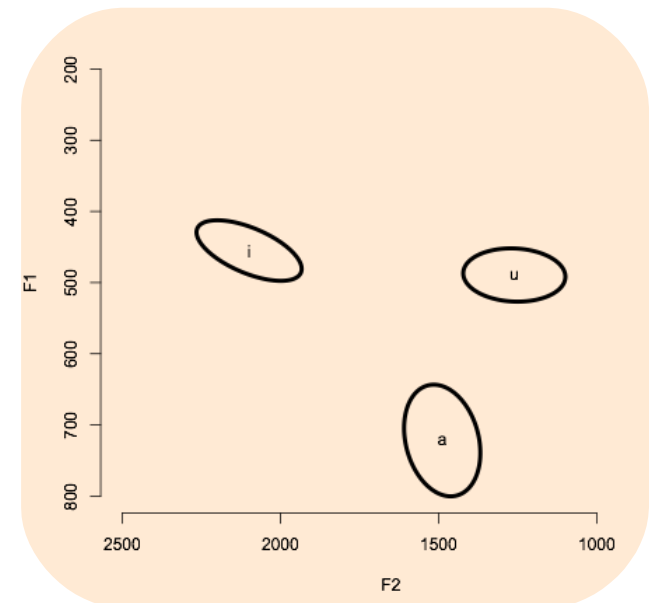
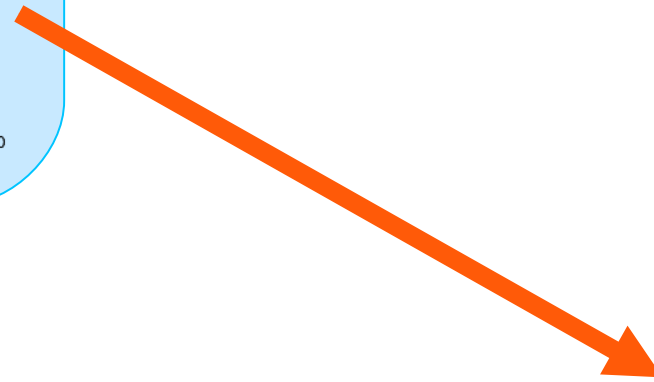
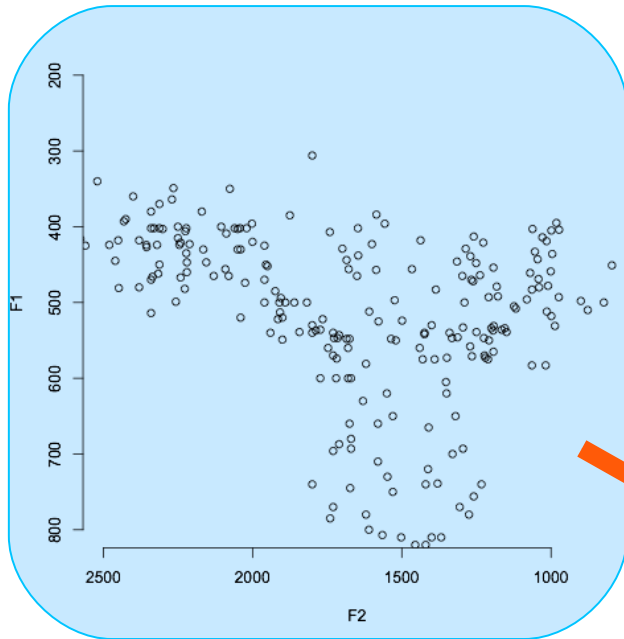
Learning Sound Categories





This diagram shows how people see the acquisition of phonemic categories: first, find the phones, then find the phonemes. This may not be an accurate representation of how everyone *wants* to think about how children discover phonemic categories, but, if only as an approximation, people tend to build models of sound category learning that either just find perceptually salient clusters (phones, e.g. Vallabha et al., 2006), or assume that we can forget about all but the possibly relevant, discrete, categories of sounds (phones), and worry about going from those to phonemic categories (e.g., most linguistic work on finding phonemes, and Peperkamp et al., 2006, the paper we're looking at).

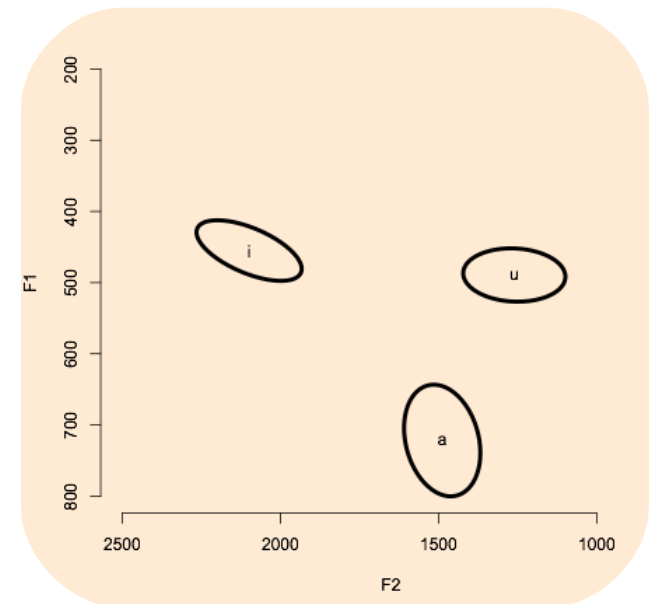
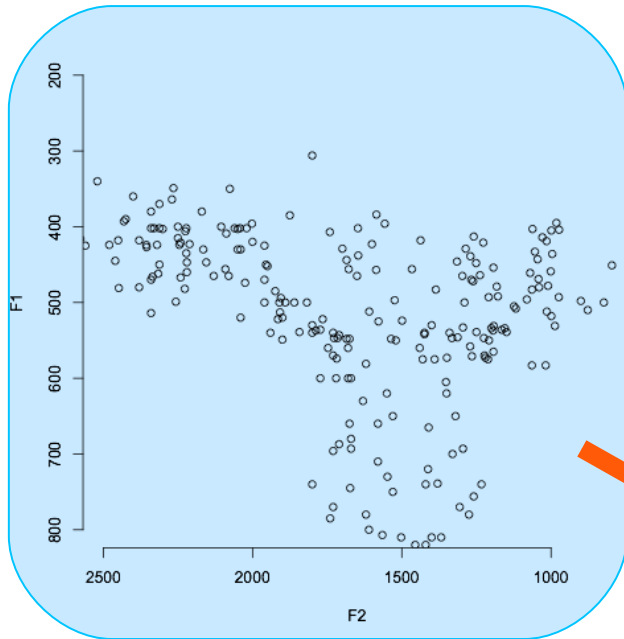
Learning Sound Categories

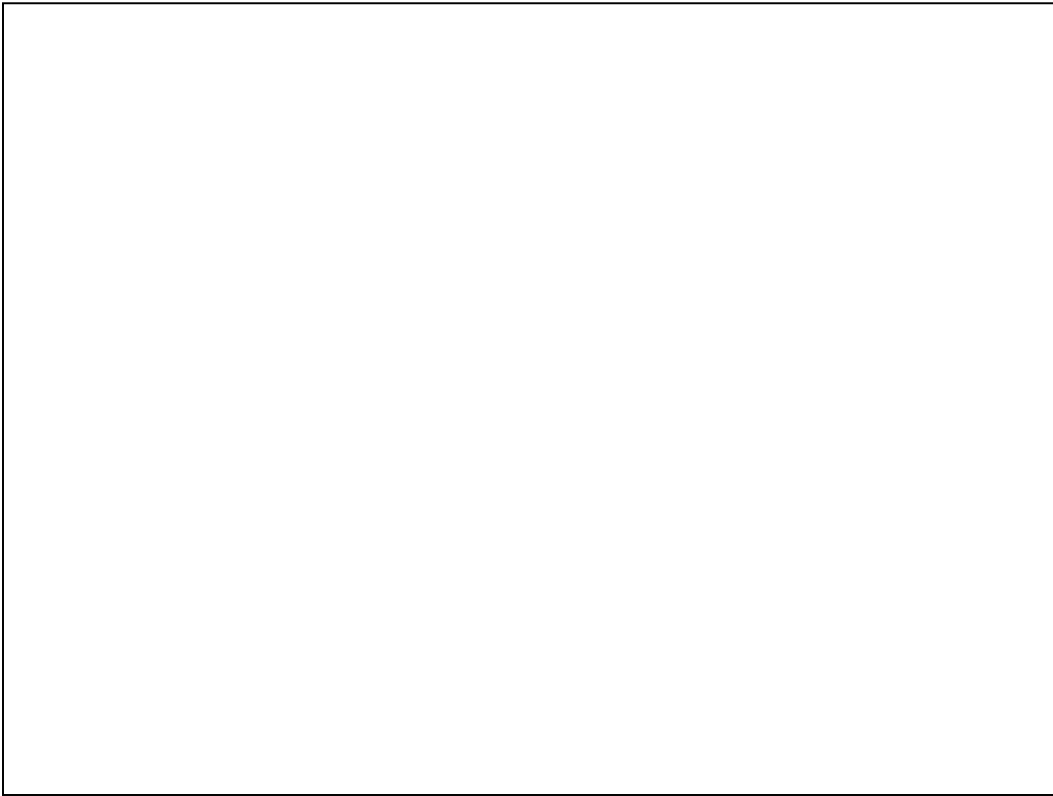




This contrasts with another story in which we can somehow jump right to the phonemes from the raw percepts . . .

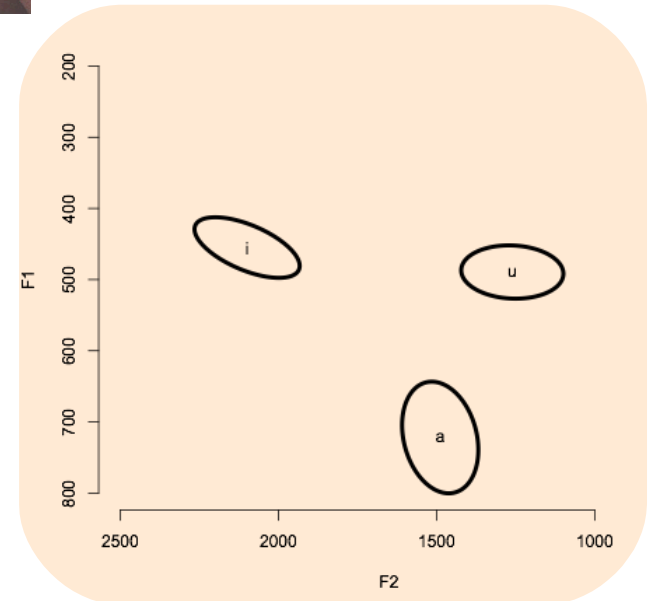
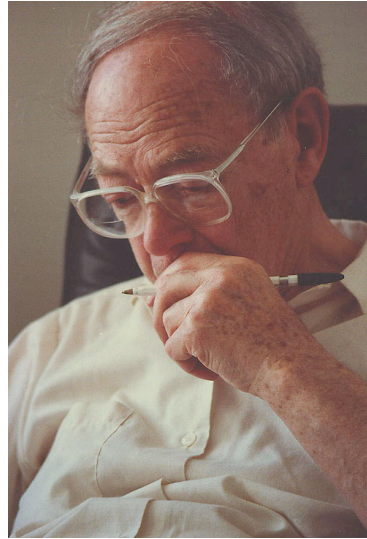
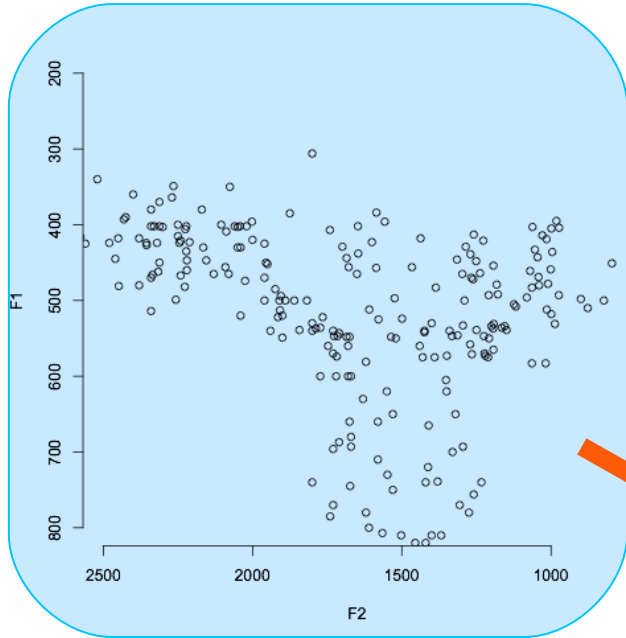
Learning Sound Categories





. . . which is a view that's very consistent with the intuitions of people who see phoneme learning as a process of making successive cuts from coarse to finer categories, since we can see the set of percepts as one big, undifferentiated category . . .

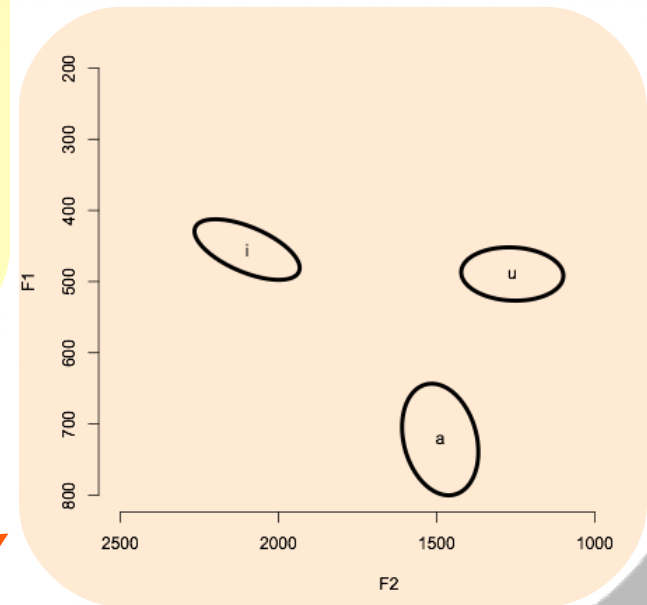
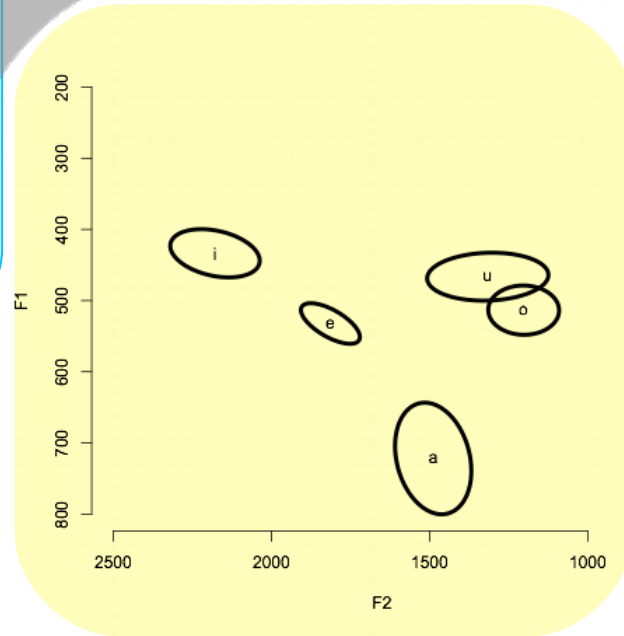
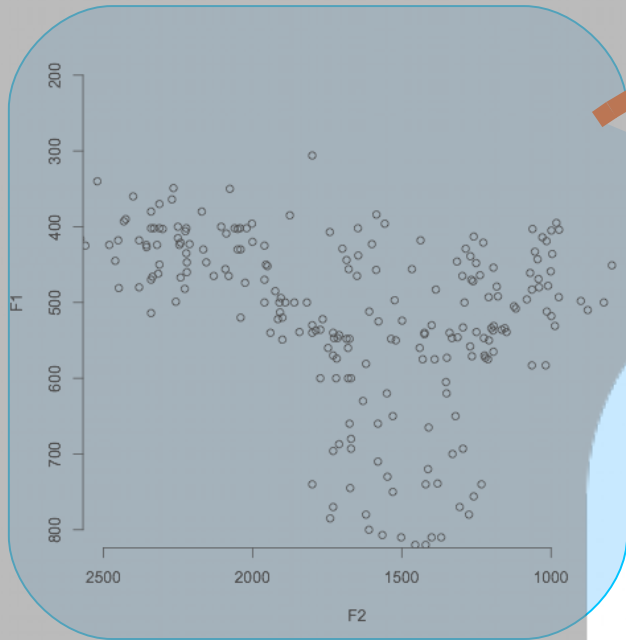
Learning Sound Categories





. . . whereas the view with the intermediate step is more consistent with Zellig Harris's "Methods," which was not about acquisition per se (it was written in the Structuralist days) but outlined procedures for linguists going from acoustics to phones (factor out free variants) and going from phones to phonemes (what we're talking about).

Learning Sound Categories





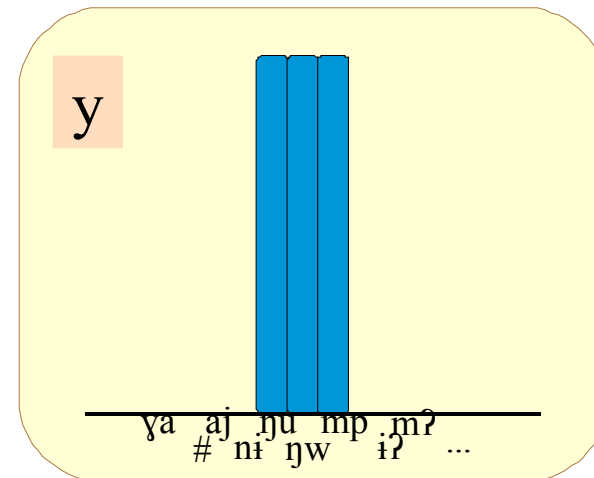
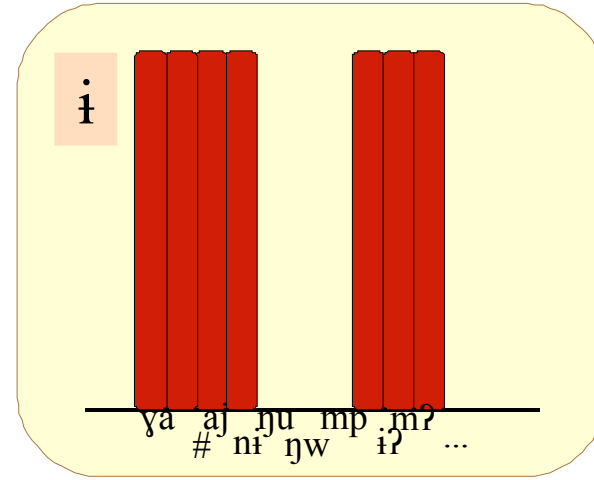
Here we focus on Harris's story about the second step, and a beefed-up version that appeared in the literature.

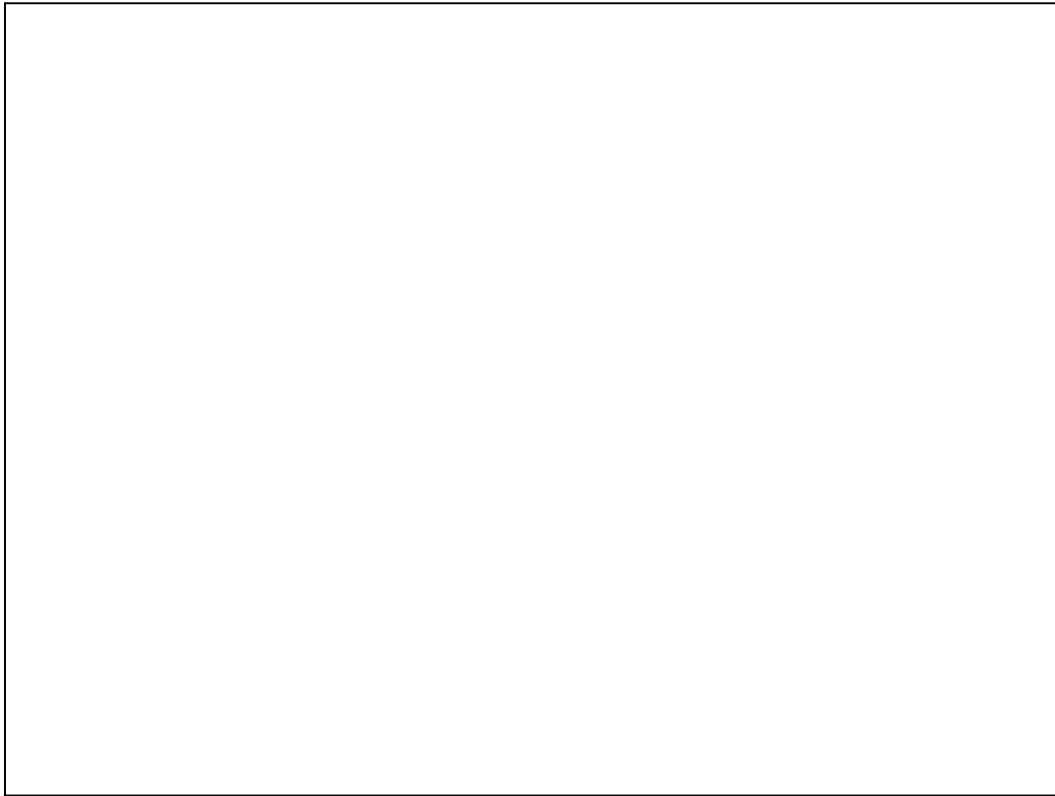
Learning Sound Categories

$i \rightarrow y / _ [nasal][labial]$

i	y
_ya	_mp
_#	_ŋw
_aj	_ŋu
_ni	
_i?	
_m?	
...	

(Southern Paiute; Sapir 1929)





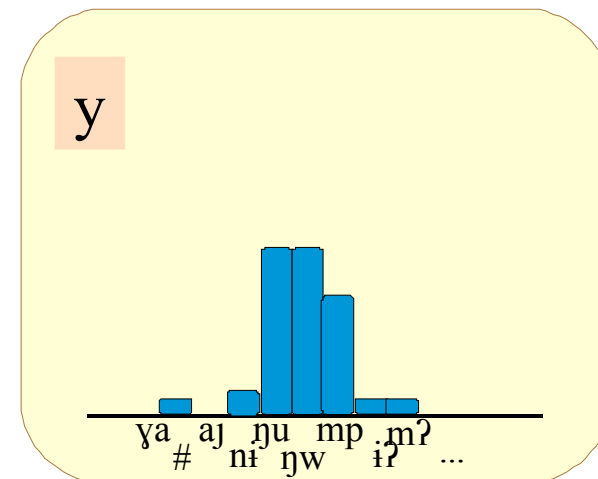
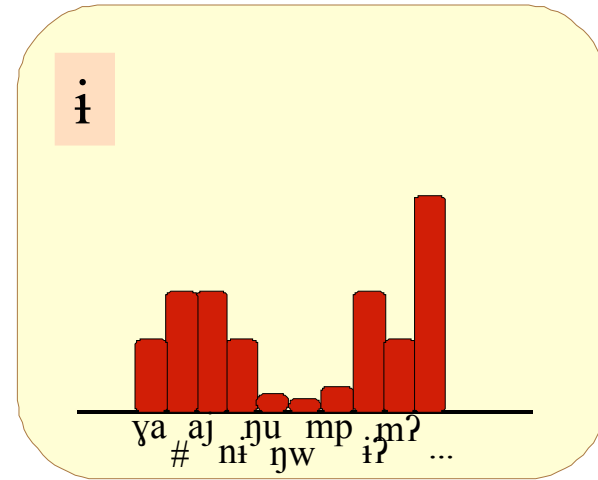
To collapse phones into phonemes, we're looking for allophonic processes. This is the classic test for allophony, and it's the one that Harris talks about. Suppose we have two phones. We can look at the *linguistic distribution* of each of the phones – the set of environments they can occur in – and see if they overlap. In the pictures at right, the blue and the red graphs should fit together. The two phones are in complementary distribution, because their distributions don't overlap. So it would be an astonishing coincidence if there wasn't a rule like the one at left.

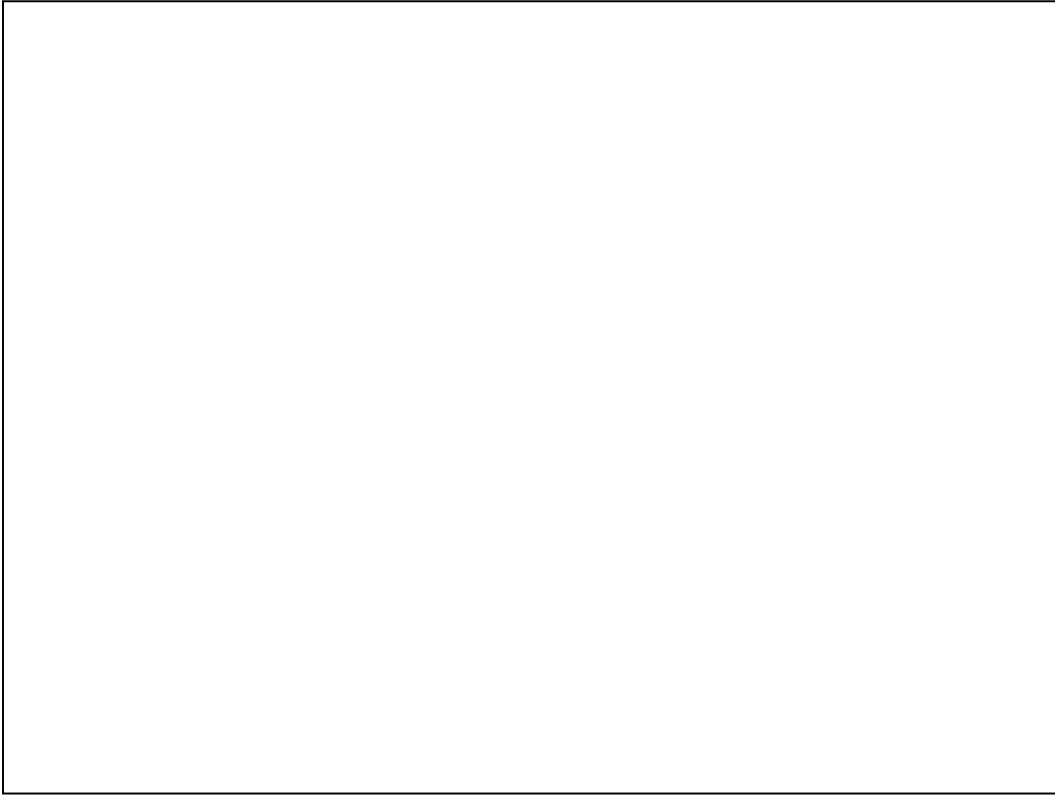
Peperkamp et al.'s Innovation

$i \rightarrow y / _ [nasal][labial]$

i	y
ya	mp
#	ɲw
aj	ɲu
ni	
i?	
m?	
...	

(Southern Paiute; Sapir 1929)





The insight of Peperkamp's group was that the *linguistic distribution of a phone* is really the *probability distribution of contexts conditioning on that phone*, as long as we also track how often each combination occurs. The intuition is that kids are good distributional learners (following the Maye et al., Saffran et al., etc line of results). So maybe they can use their agility with probability to find allophones and thus collapse (some) things into phonemes.

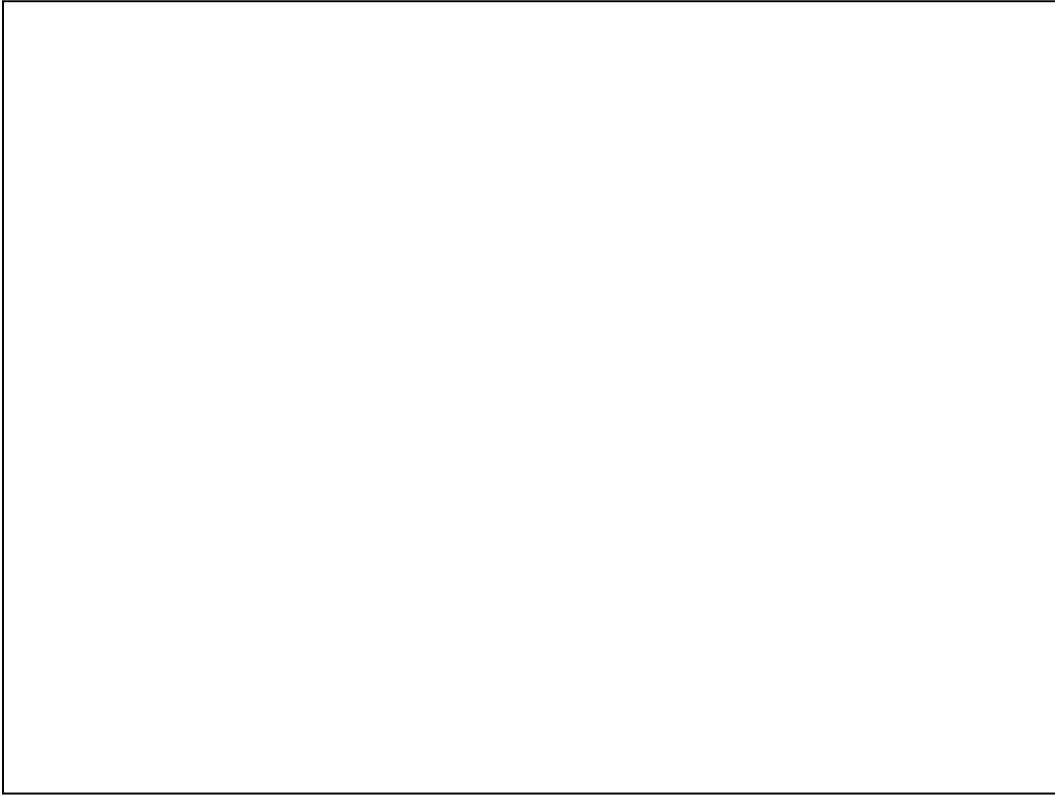
KL-Divergence

Kullback and Leibler 1951: quantify the **difference between two probability distributions** by

$$\sum_{\mathbf{x}} \Pr_{\mathbf{P}}(\mathbf{x}) \log \frac{\Pr_{\mathbf{P}}(\mathbf{x})}{\Pr_{\mathbf{Q}}(\mathbf{x})} + \sum_{\mathbf{x}} \Pr_{\mathbf{Q}}(\mathbf{x}) \log \frac{\Pr_{\mathbf{Q}}(\mathbf{x})}{\Pr_{\mathbf{P}}(\mathbf{x})}$$

Symmetrized Kullback-Leibler Divergence

Peperkamp et al. 2006, 2007; Calvez et al. 2007: pairs of segments with **context distributions with large KLD** (very different) are **likely allophones**



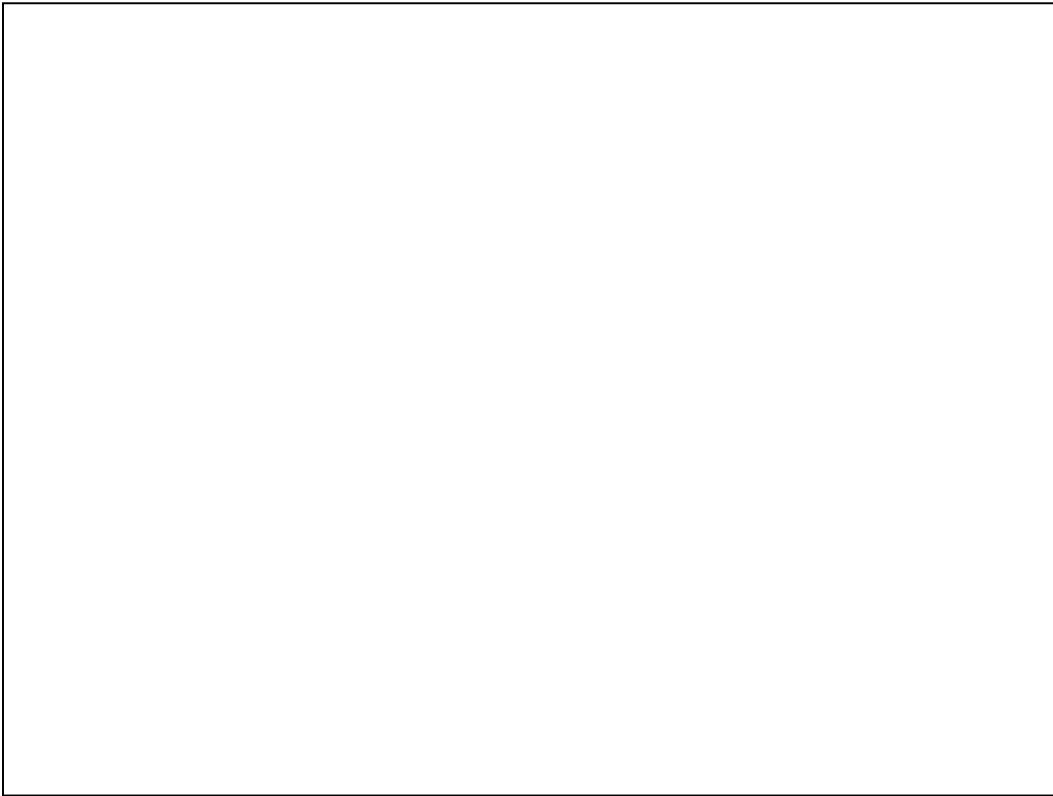
So here is how they are going to cash this out: the *Kullback-Leibler divergence* (KLD), a quantity from information theory that's also called *relative entropy*. This quantity gets bigger as two probability distributions get more different. Skip ahead to the example if you don't need to know the details of how this works . . .

KL-Divergence

Kullback and Leibler 1951: quantify the **difference** between two probability distributions by

$$\sum_{\mathbf{x}} \Pr_{\mathbf{P}}(\mathbf{x}) \log \frac{\Pr_{\mathbf{P}}(\mathbf{x})}{\Pr_{\mathbf{Q}}(\mathbf{x})}$$

Kullback-Leibler Divergence (Relative Entropy)



. . . so, in fact, the quantity Peperkamp et al. call KL divergence, most people wouldn't call KL divergnece. For most people KL divergence is not symmetrical, so you have the KLD of probability distribution P against probability distribution Q, which is not the same as the KLD of Q against P.

KL-Divergence

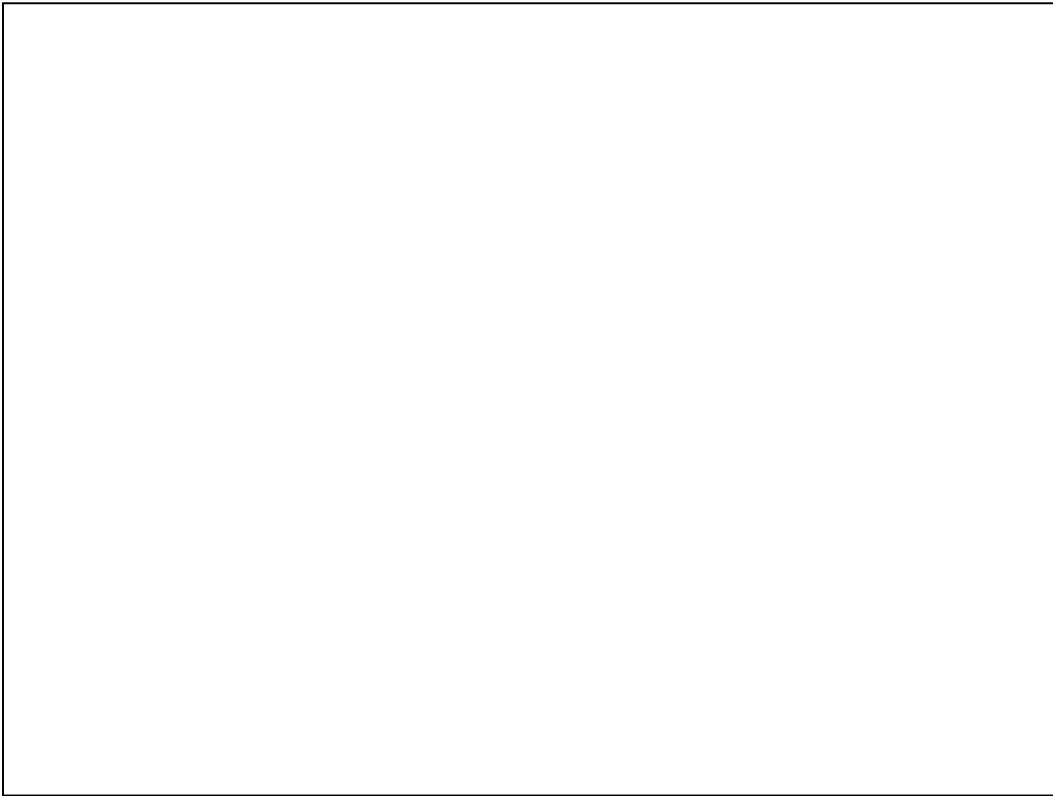
Kullback and Leibler 1951: quantify the **difference** between two probability distributions by

$$\sum_x \Pr_P(\mathbf{x}) \log \Pr_P(\mathbf{x}) - \sum_x \Pr_P(\mathbf{x}) \log \Pr_Q(\mathbf{x})$$

-(Entropy of P) plus Cross-Entropy of P under Q

Average number of bits needed to encode an event from P using an optimal coding scheme

Average number of bits needed to encode an event from P using a coding scheme that would be optimal for Q-distributed events



That quantity, by properties of summation and logs, is the cross-entropy of Q given P minus the entropy of P . Entropy is the theoretical optimum for the smallest number of bits we would need to encode some average message if we know the probability distribution of possible messages (you can show that that if you know how likely various messages are, you can get away with using fewer bits for more frequent things). Cross-entropy is just subbing out some wrong probability distribution for the real probability distribution in this same case (in which case you can imagine you'd have to use some sub-optimal number of bits to encode messages if you thought that the frequent messages were common and vice versa). Subtract the entropy from the cross-entropy and you get the number of extra bits you use to encode messages from P if you think they are distributed as Q . That's the standard use of "KL Divergence."

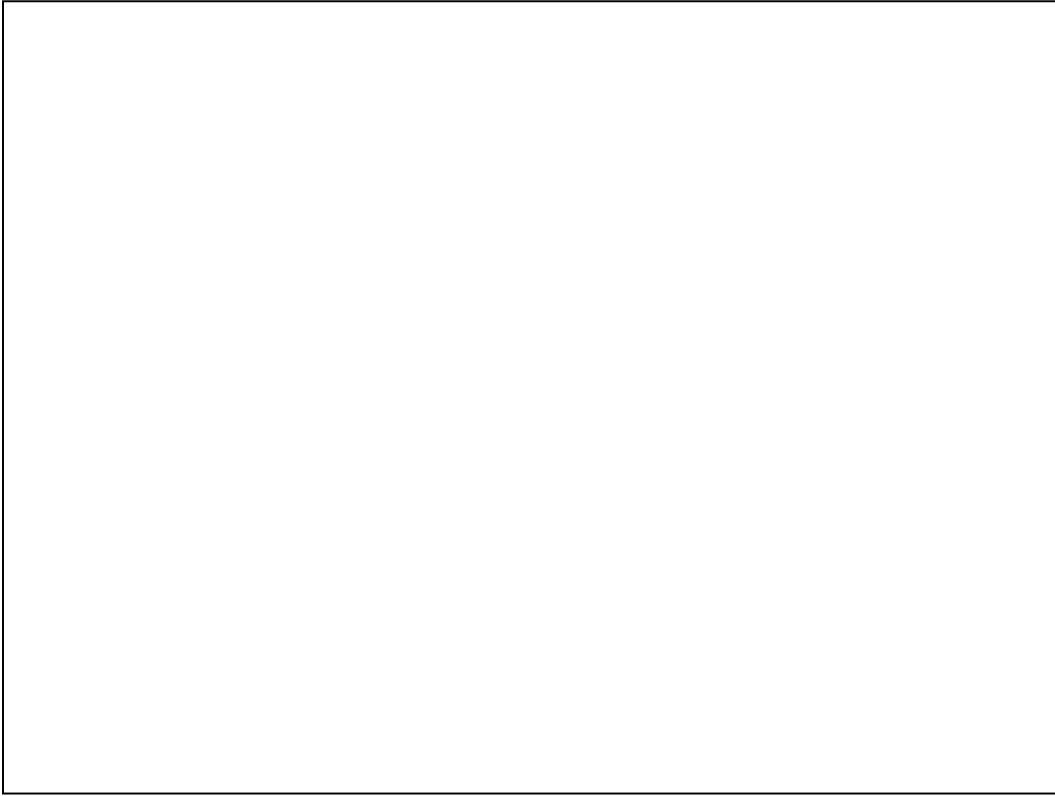
KL-Divergence

Kullback and Leibler 1951: quantify the **difference** between two probability distributions by

$$\sum_{\mathbf{x}} \Pr_{\mathbf{P}}(\mathbf{x}) \log \frac{\Pr_{\mathbf{P}}(\mathbf{x})}{\Pr_{\mathbf{Q}}(\mathbf{x})} + \sum_{\mathbf{x}} \Pr_{\mathbf{Q}}(\mathbf{x}) \log \frac{\Pr_{\mathbf{Q}}(\mathbf{x})}{\Pr_{\mathbf{P}}(\mathbf{x})}$$

Symmetrized Kullback-Leibler Divergence

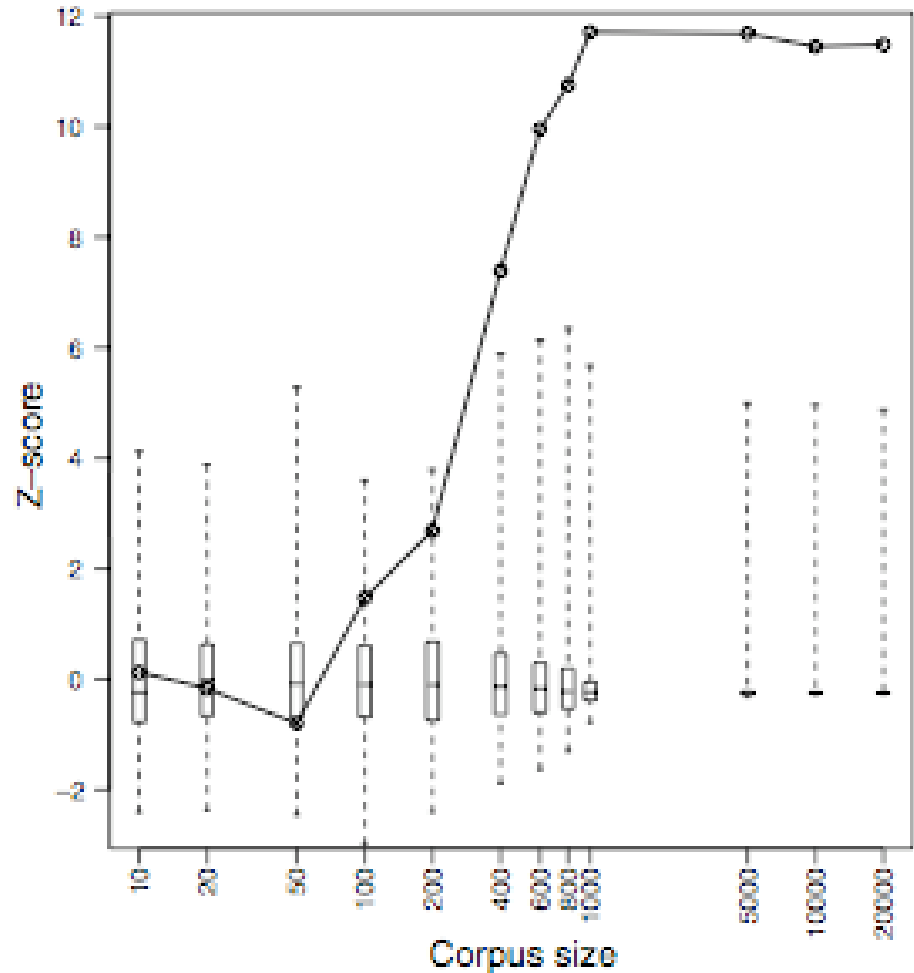
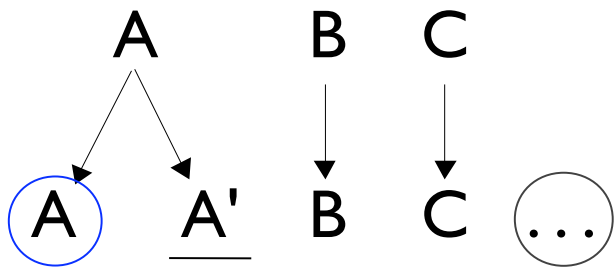
Number of extra bits we need to encode things from distribution Q if we think they're things from distribution P plus the other way round.

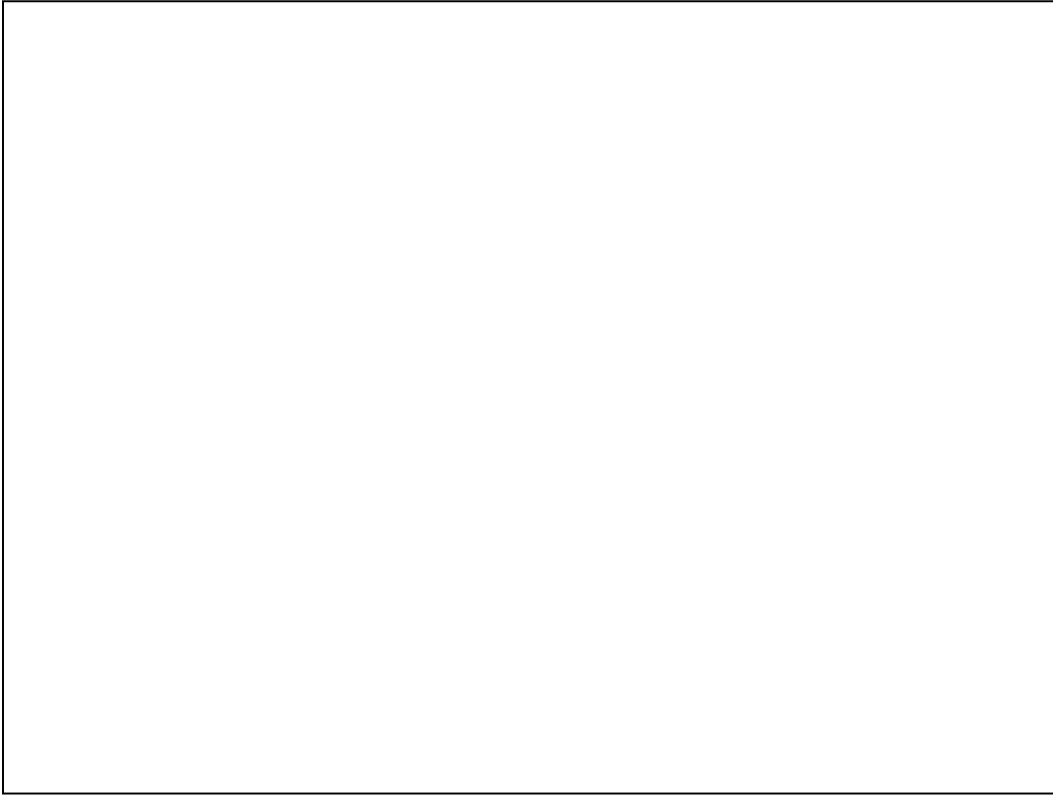


To make their job easier, Peperkamp et al. use the sum of the two KL Divergences to give a kind of “discrepancy” figure for the two distributions.

Example

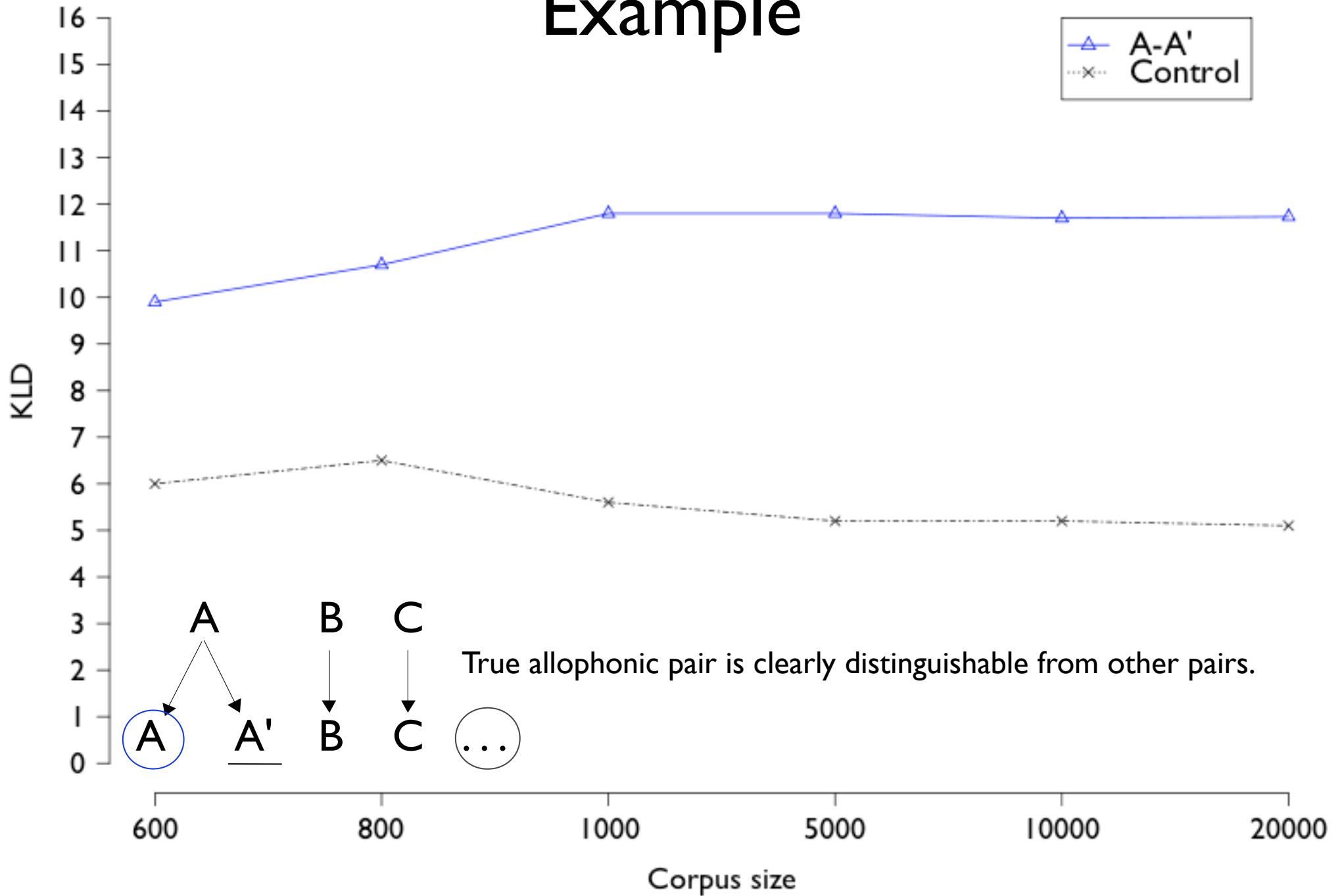
- Artificial language consisting of randomly generated strings from an alphabet of 46 “phonemes”
- One phoneme (A) changes to an allophonic variant (A') in exactly eight contexts
- Measure KL divergence between A and all other segments on various corpus sizes

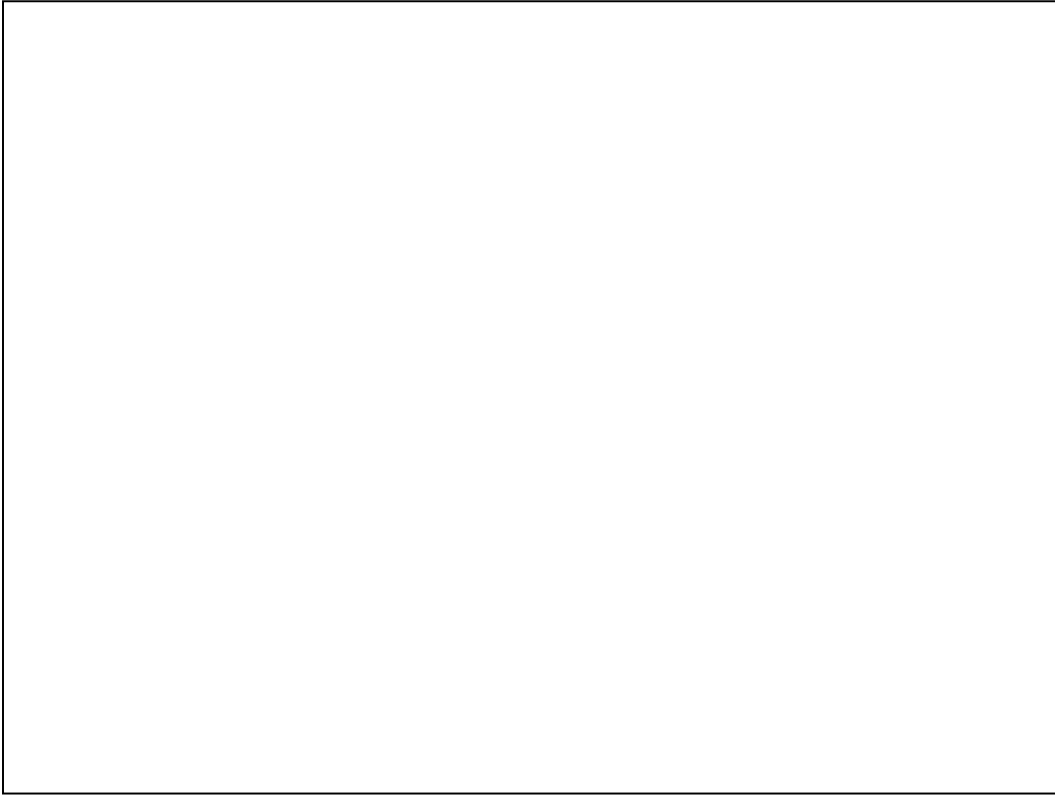




With that in mind, let's turn to an example. Peperkamp et al. run two experiments. I'm going to talk about their first, and later I'll briefly mention why the second is far less important than they claim. Their first experiment looks like this: cook up an artificial language by generating strings of characters (“phonemes”) and then apply an allophonic rule to one of the phonemes. Their allophonic rule takes A to A' (*not* an independent phoneme) in eight contexts. Now they take a bunch of these corpora with more and more examples in them, then, for each corpus, find the KLD of *every pair of phonemes*, normalize the scores over that experiment, and plot all the scores. Their box plots are a little confusing. So far as I can tell the ends of the whiskers are the largest and smallest standardised (Z) scores, except after the bold line crosses. The bold line is the A-A' Z-score, which is the one we want to be big; so right of the line the top of the top whisker is the second largest Z-score. But it's only the bold line and the top whisker we're interested in. The bold line is the comparison we want, the top whisker is the next-largest KLD score (which, I can assure you, is going to be a comparison between A' and something else, since A' has a very different distribution from *everything*). I have replotted this to show you the way I do these graphs . . .

Example

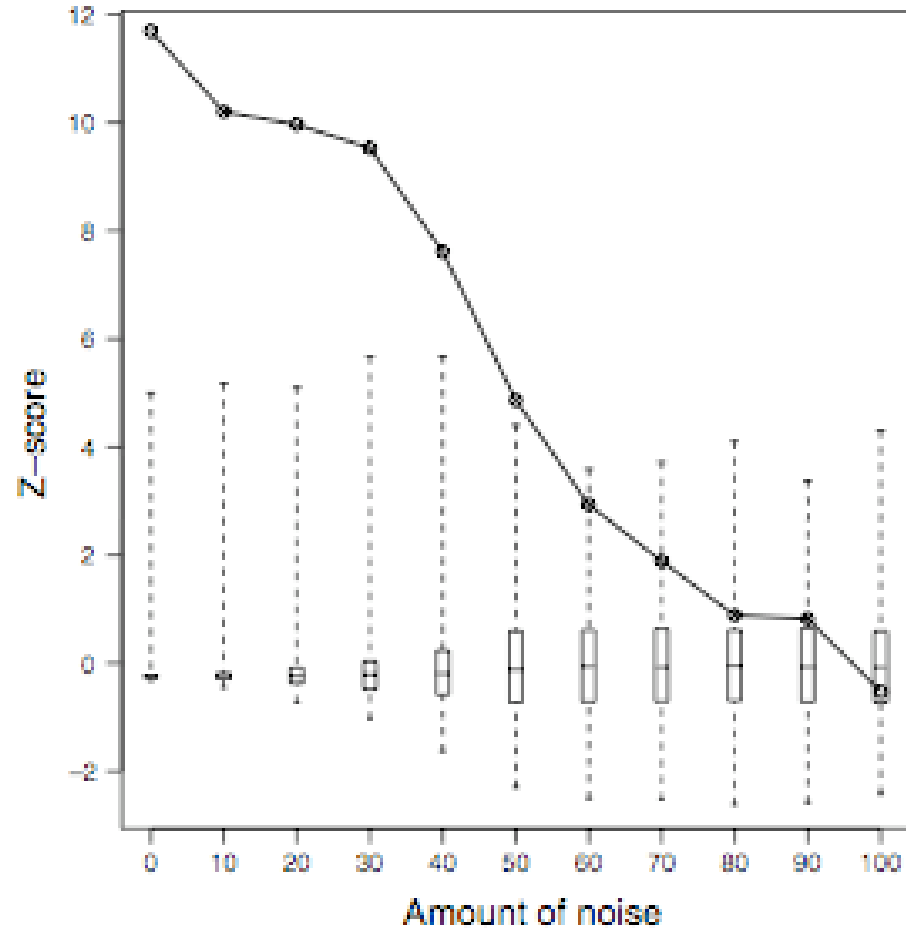
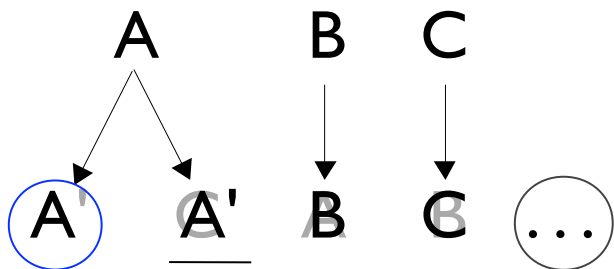


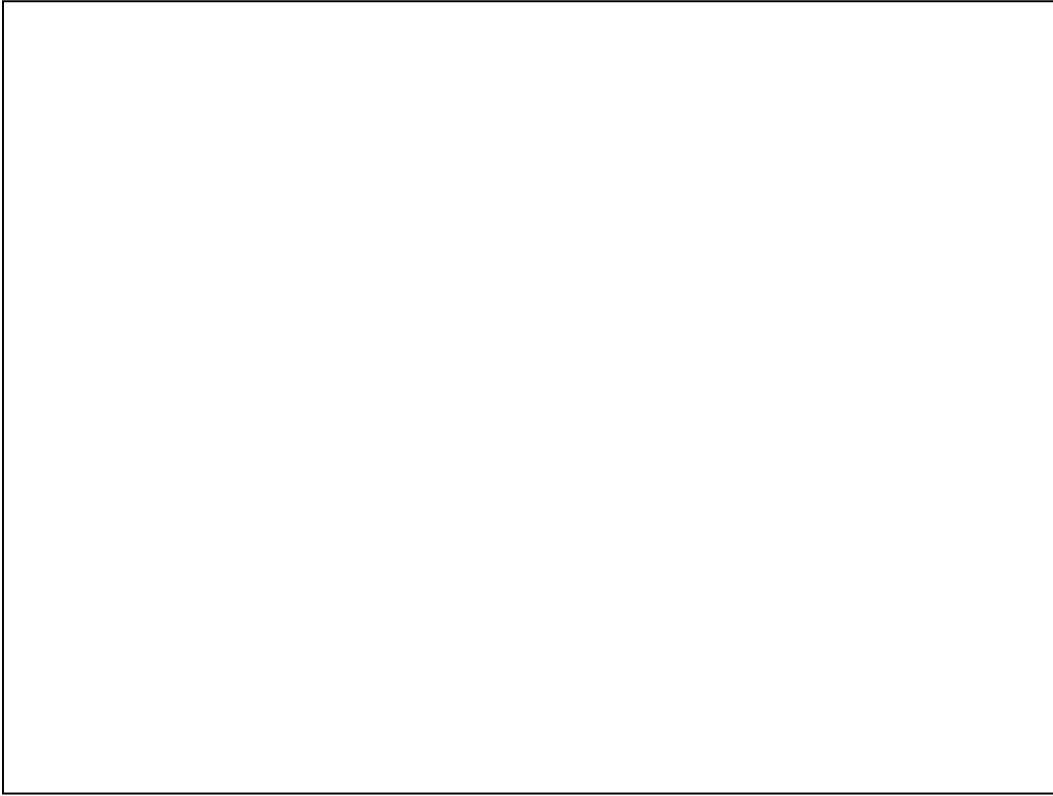


This is the second half of their graph. The blue line is the one to watch, the grey line is the top whisker. It's our control. We need to beat that if we want to claim that we can use these scores to find allophones. And, of course, we can. We could draw a line through at some predetermined level (maybe 7) and find that all the points above that line are allophonic pairs, all the comparisons below are not.

Example

- Artificial language consisting of randomly generated strings from an alphabet of 46 “phonemes”
- One phoneme (A) changes to an allophonic variant (A') in exactly eight contexts
- Add noise (randomly replace some percentage of segments)

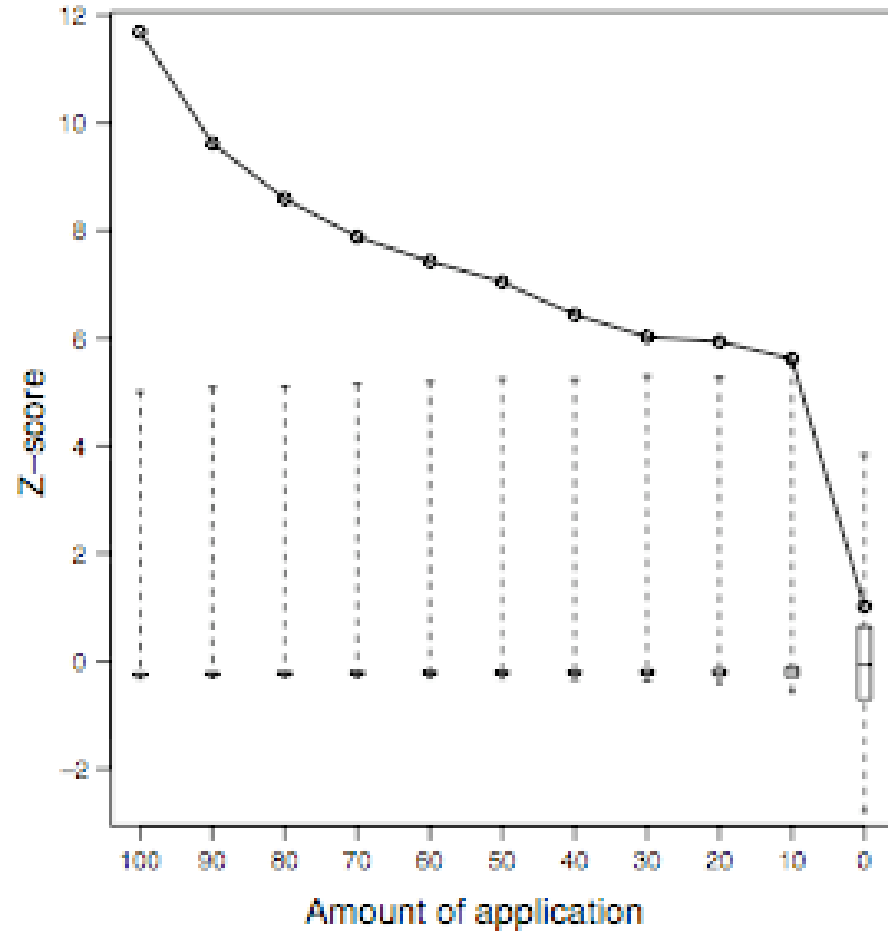
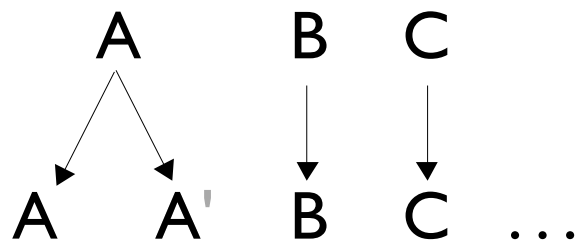


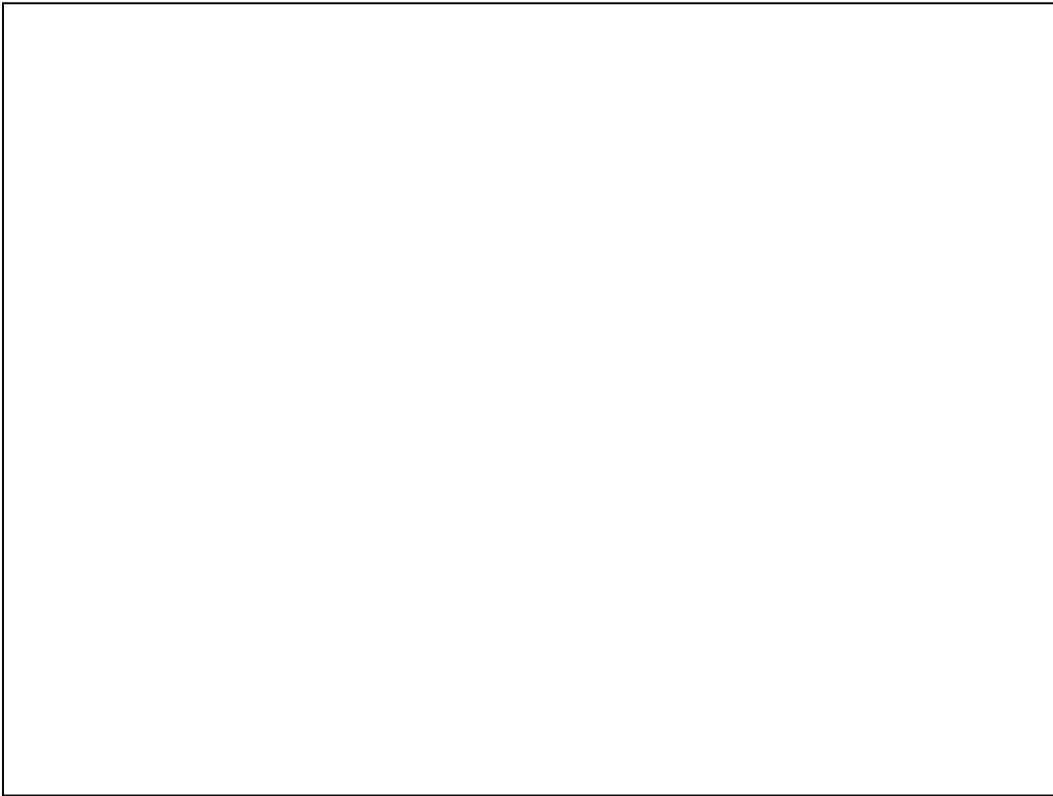


The neat thing is their second part of this first experiment. They do the same thing, but then, after the rules apply, they randomly change X% of the segments to random other segments. This is going to *obscure* the complementary distribution, so that Harris will fail. But they will succeed, because KLD is a strict *generalization* of complementary distribution. All complementary distributions have large KLDs, not all large KLDs have complementary distributions, including some of the allophonic comparisons we're going to miss if we require strict complementary distribution. So, throw in some noise, and we still get the right result, up to about 40% noise.

Example

- Artificial language consisting of randomly generated strings from an alphabet of 46 “phonemes”
- One phoneme (A) changes to an allophonic variant (A') in exactly eight contexts
- The allophonic rule applies variably (Labov)





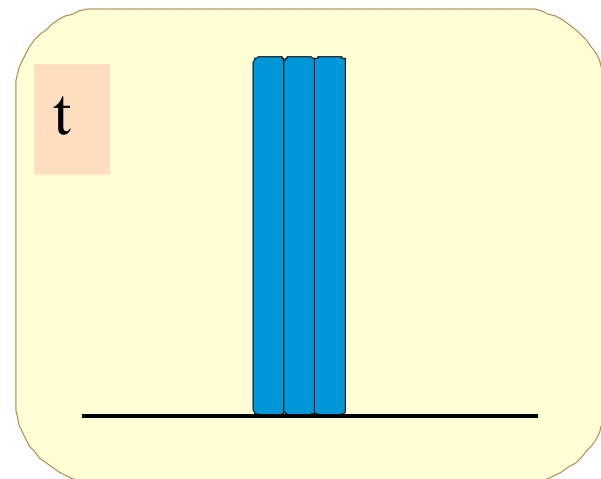
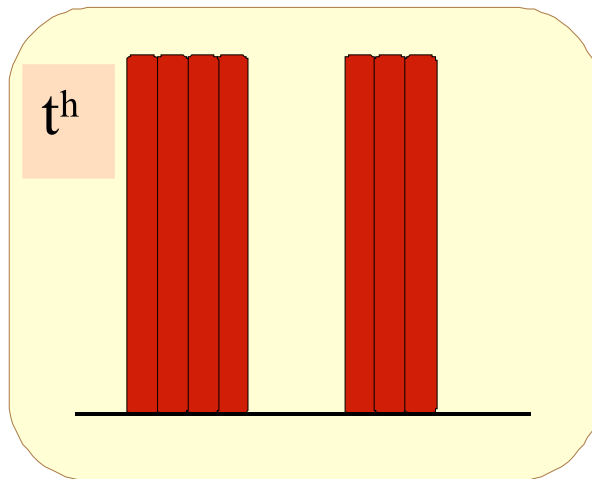
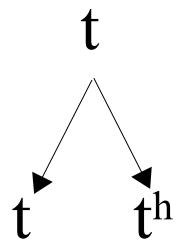
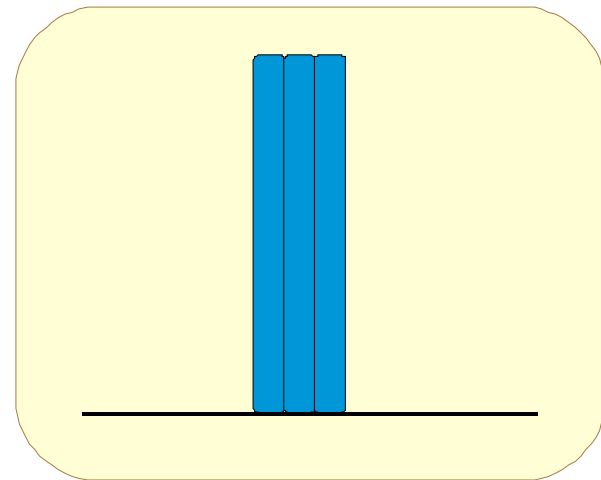
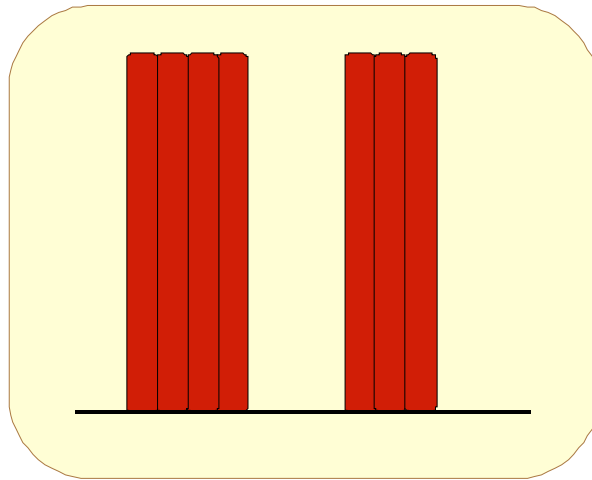
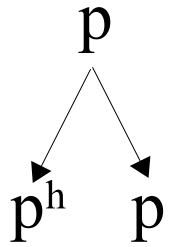
And their third part in the artificial language experiment is the “socio” experiment. Can you still find rules when they're inconsistently applied? Yes. So the neat thing here is, as people have demonstrated time and time again (eg. Yang 2002), statistical algorithms are great and probably crucial for filtering out noise in the data. But here's the rub . . .

Not All Noise Is Random

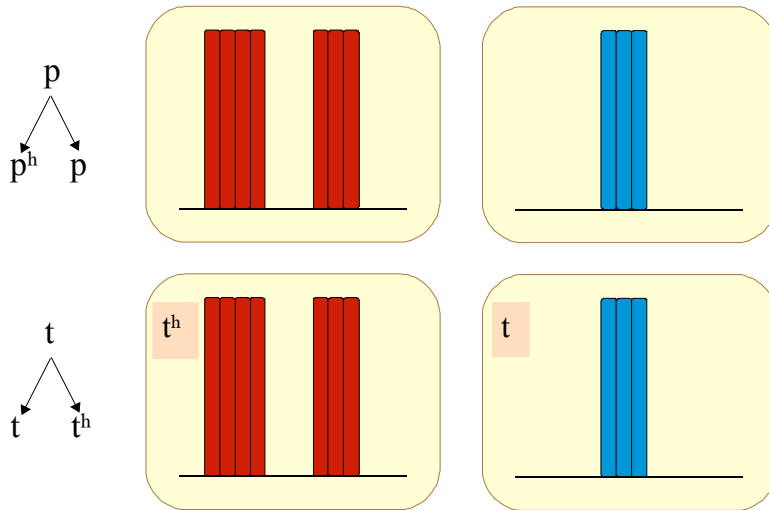


Peperkamp's algorithm is an extension of Harris's. But every serious linguist knows that complementary distribution is full of holes.

Harris's Problem

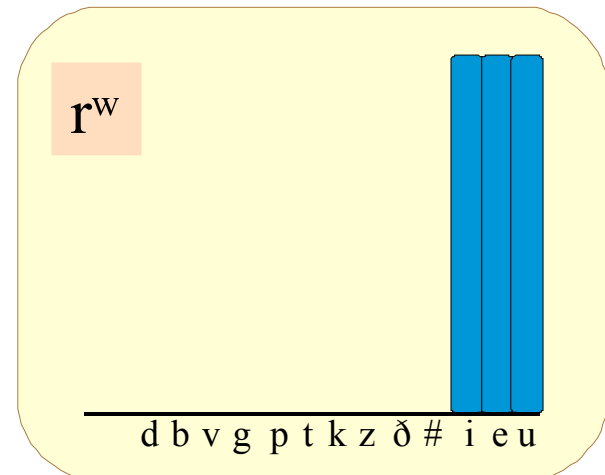
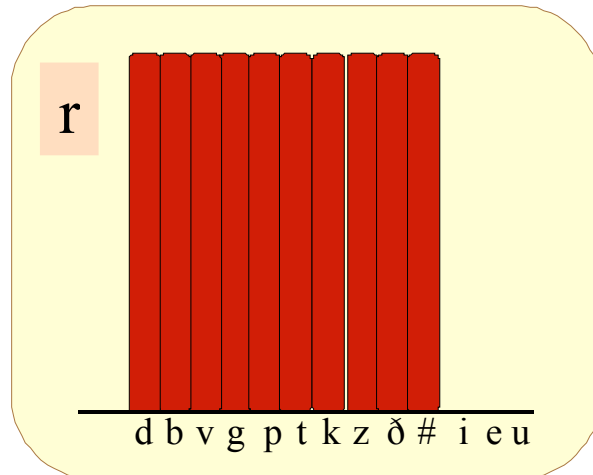
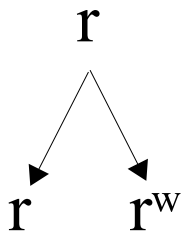
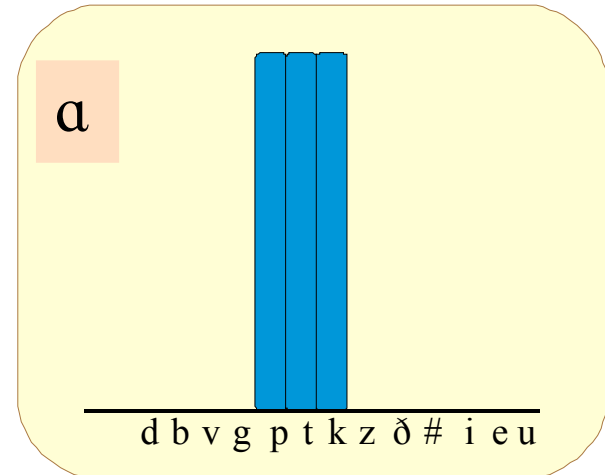
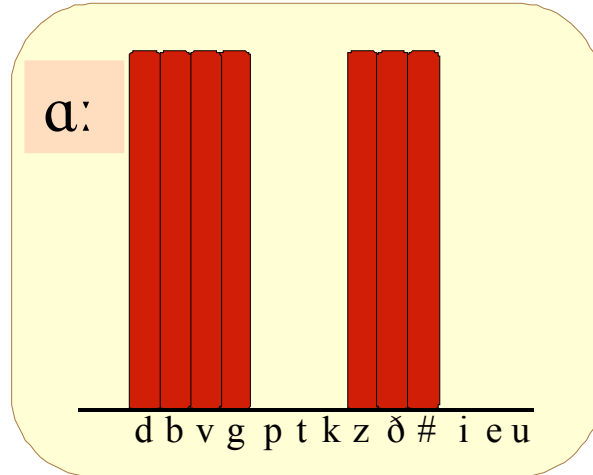
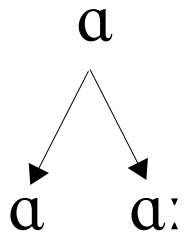


Harris's Problem

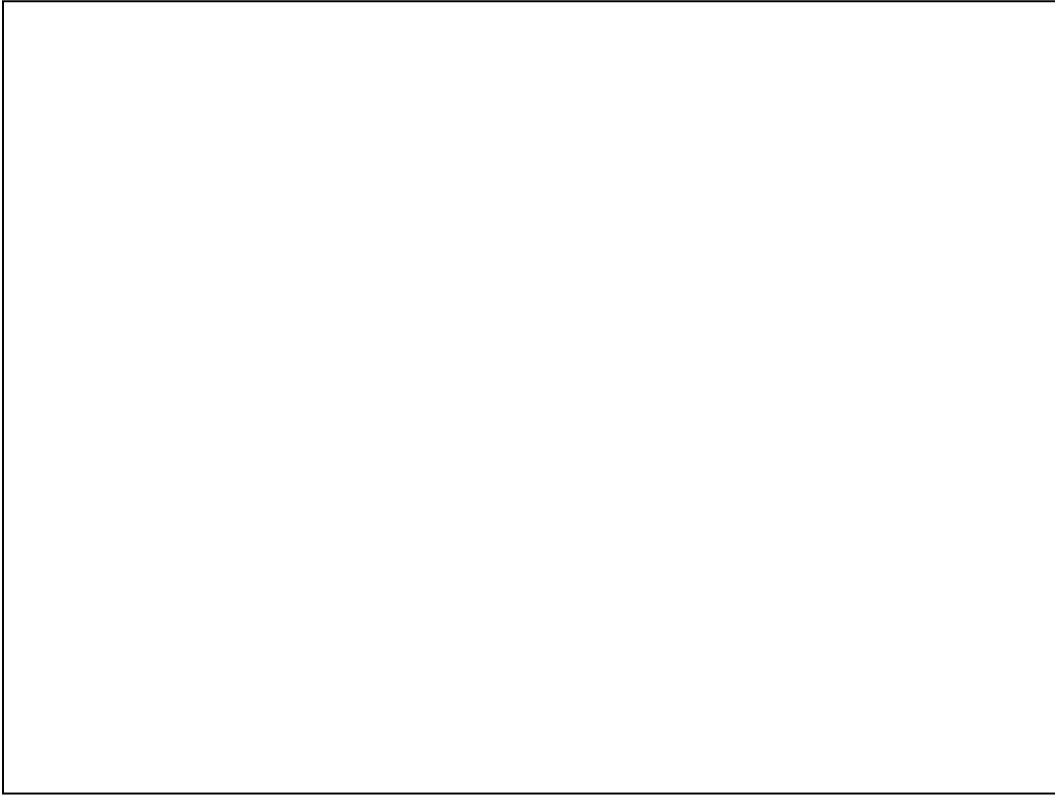


So, here's an example you'll find if you just keep reading Harris's book. In the classic case of English aspiration, we find that the complementary distribution test totally underdetermines the right answer. So the comparison between *ph* and *p* or between *th* and *t* is *no different than the comparison between ph and t or th and p*. All these things have complementary distributions. Oops. Harris's solution is to say that you look for parallel pairs of allophones (or, interpolating, that you prefer to have a single rule). I won't show how Peperkamp's algorithm does in this case (it fails) but keep in mind Harris's solution.

Multiple Independent Processes



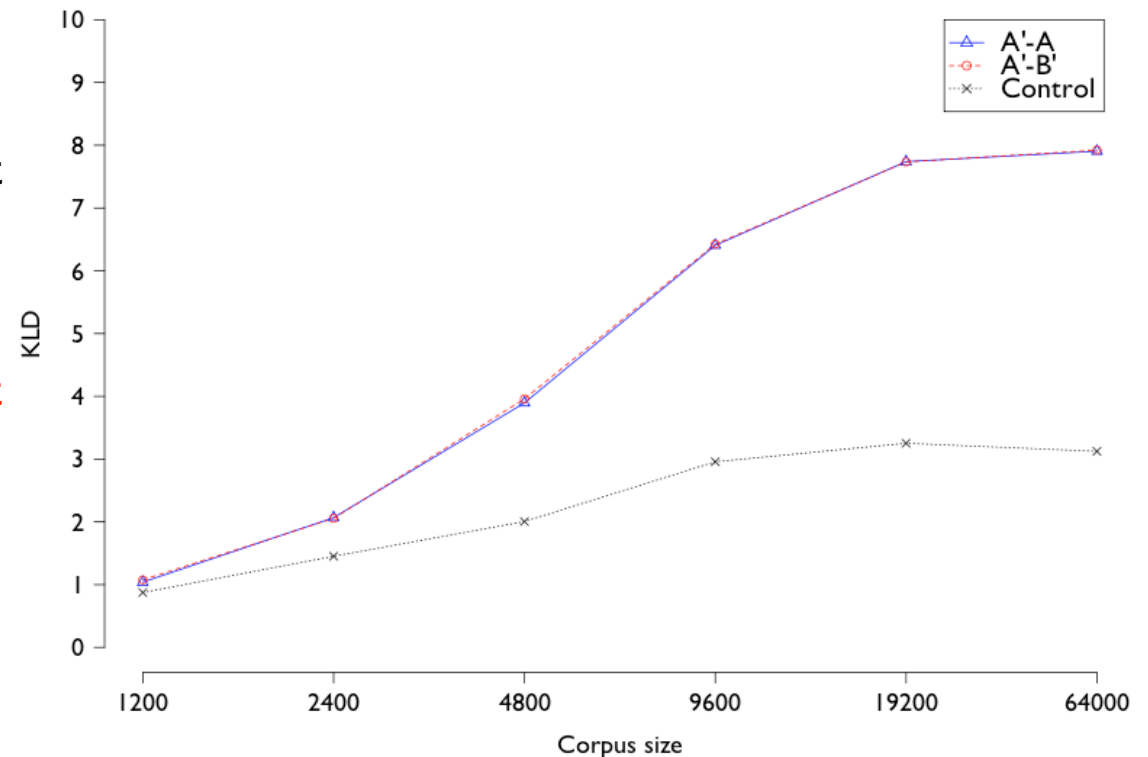
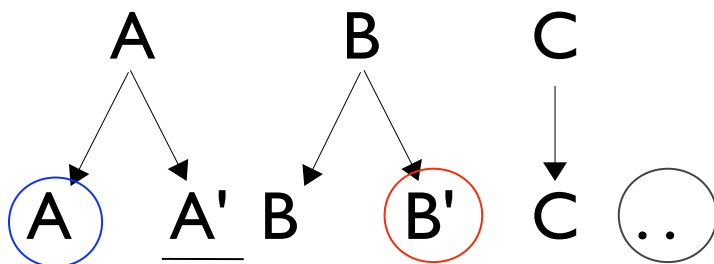
(English)

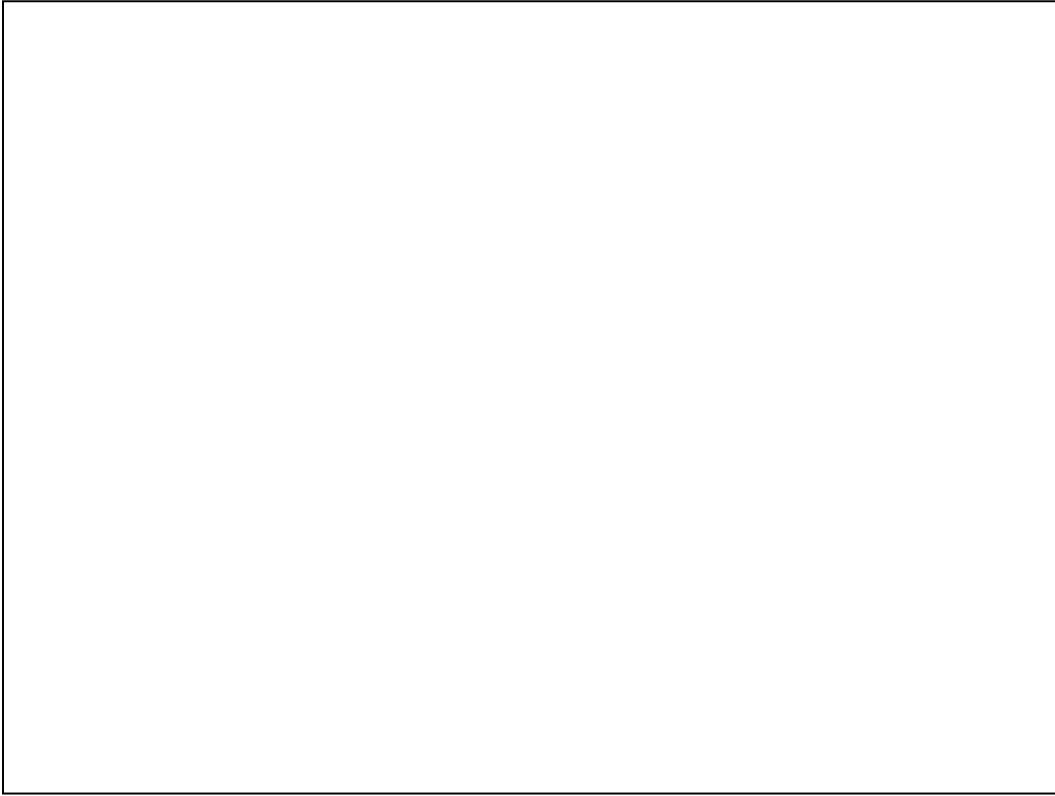


Here's a similar case which I will show you a run of Peperkamp's algorithm on. In English, we have complementary distribution between long and short vowels (to some idealization). There's also a rule of r-rounding, at least in North American dialects, before vowels (though again the phonetics are messy). Suppose you notice all these phones. Then if you use the Harris/KLD test, you find the complementary distribution between a and a:, and the complementary distribution between r and rw, and you're done, right? No... look more closely. What about a versus rw?

Multiple Independent Processes

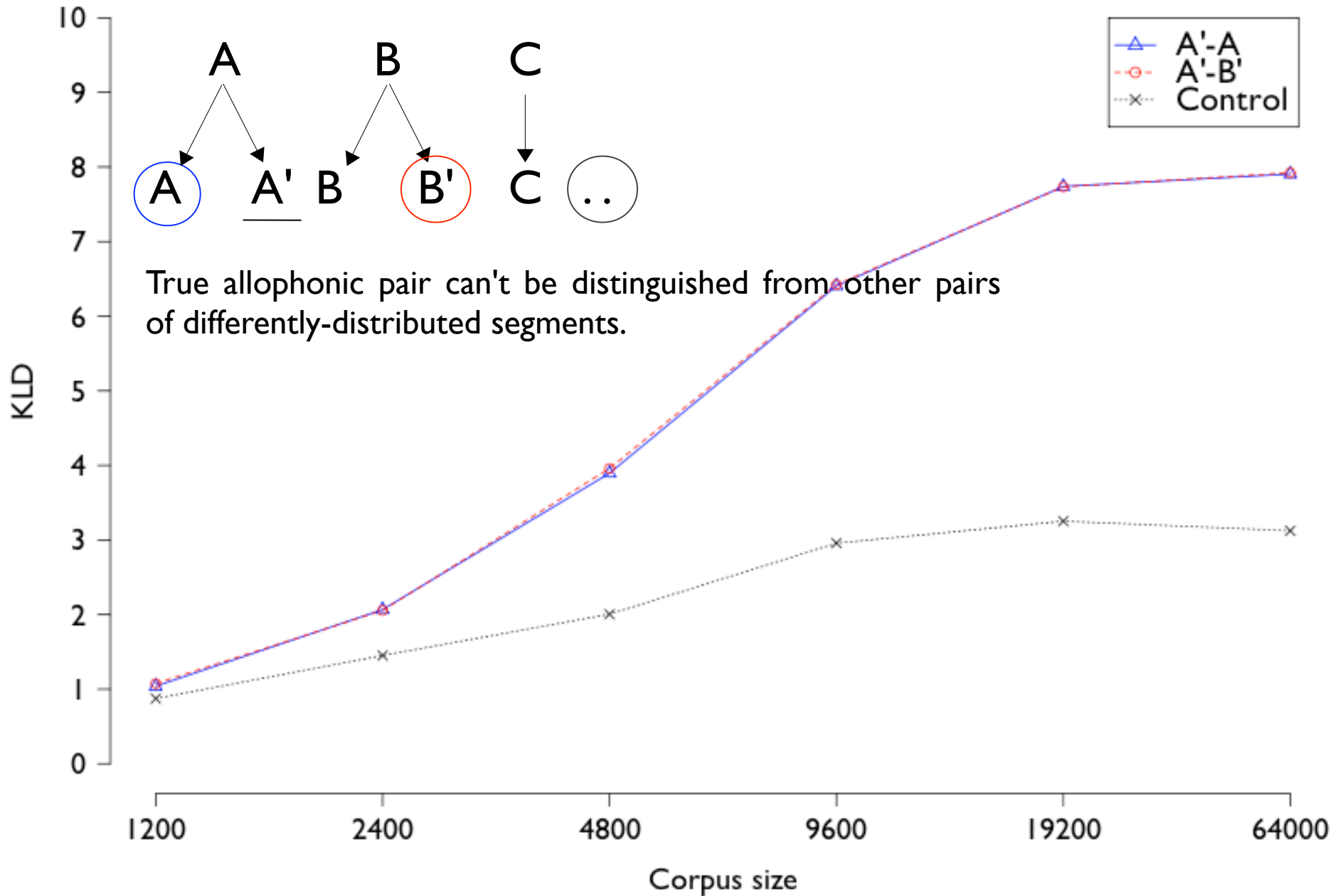
- Artificial language consisting of randomly generated strings from an alphabet of 46 “phonemes”
- One phoneme (A) changes to an allophonic variant (A') in exactly eight contexts
- Another phoneme (B) changes to an allophonic variant (B') in eight contexts, disjoint from those in which A' occurs

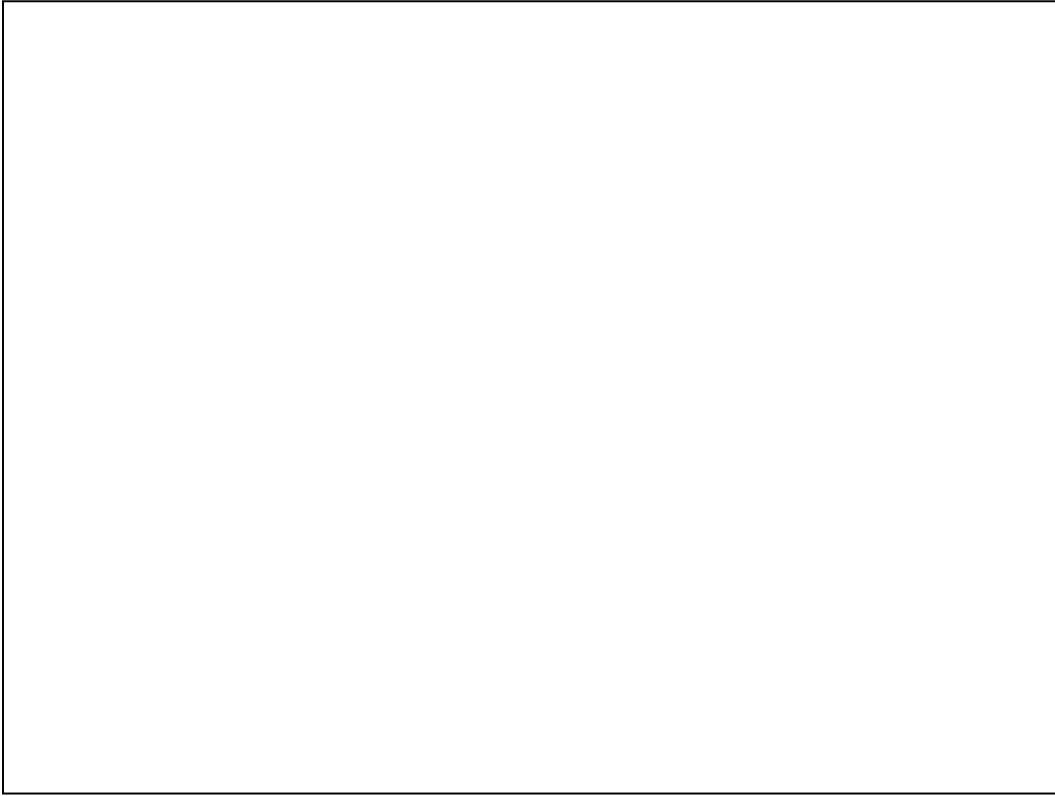




Let's run this in Peperkamp's framework. Let's do their experiment again, and then throw in a second rule. That's all we're going to do. So we're leaving the set of phonemes the same, and we're adding an extra phone, B', occurring in eight contexts, crucially, totally distinct from the contexts in which we get A'. And now let's compare the A-A' KLDs with the A'-B' (spurious) KLDs . . .

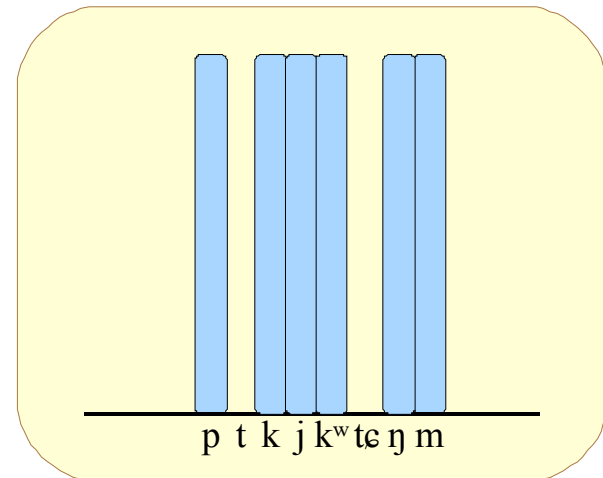
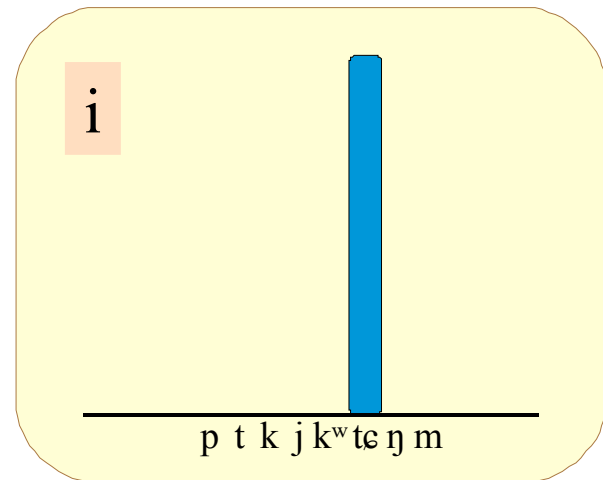
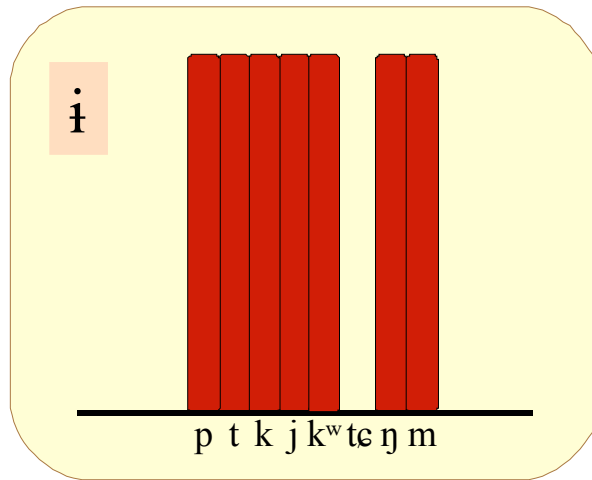
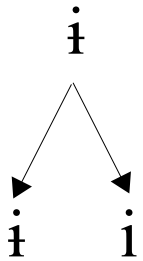
Multiple Independent Processes



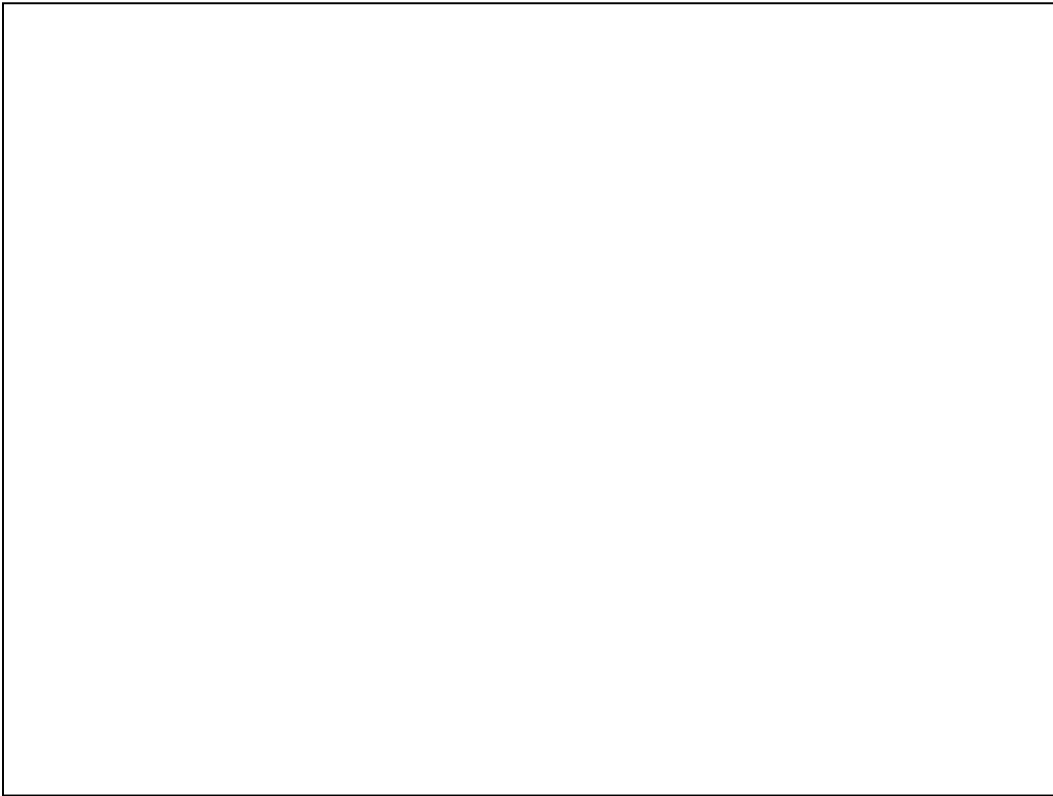


Here we're comparing them both to the grey line (baseline – largest other KLD score). Can you even see that there are two lines there? No. They overlap pretty much exactly. This is not a mistake! We can't separate the spurious from the real allophonic pair. Peperkamp's algorithm fails here. (But what about Peperkamp et al.'s second experiment? Hold that thought . . .)

Neutralization



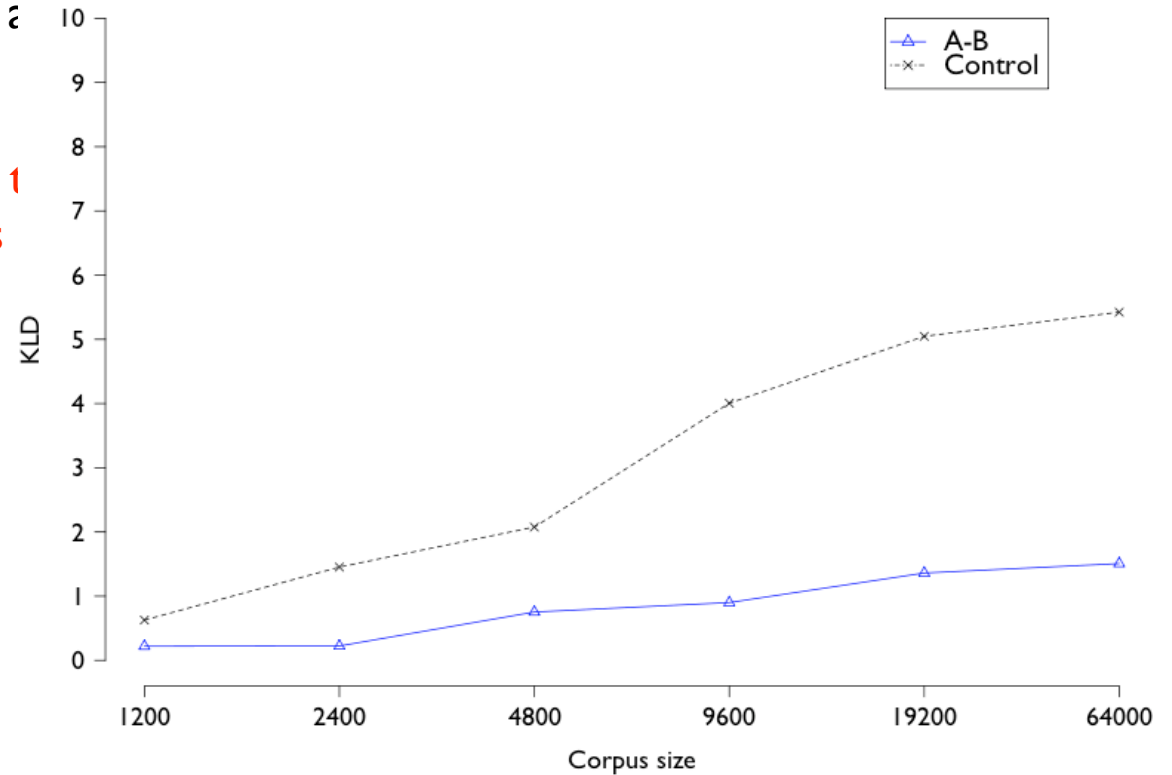
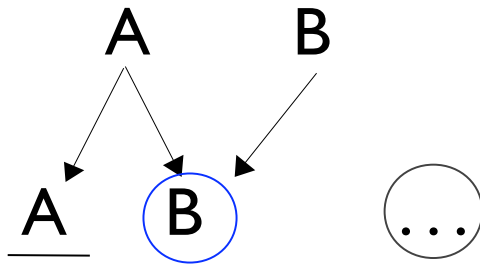
(Southern Paiute; Sapir 1929,
Lovins 1972)

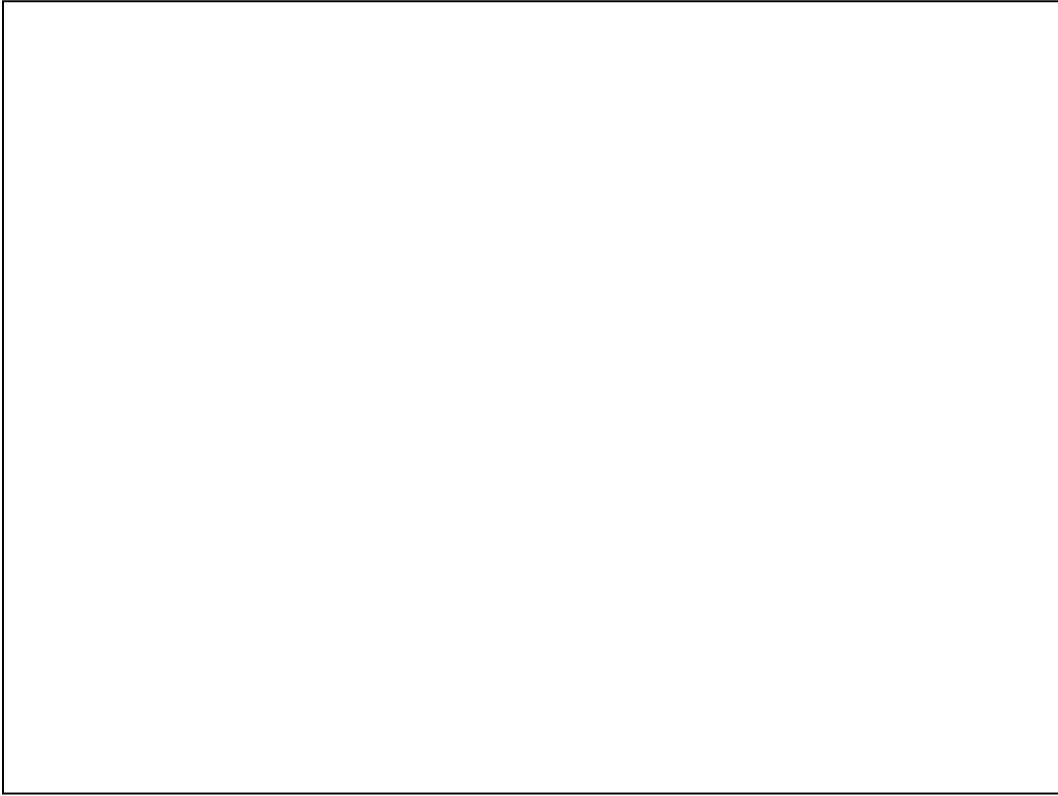


Now, Peperkamp et al. published a paper about finding allophones. But every undergrad knows that complementary distribution can *only* find strict allophones, rules that are *non* structure preserving. We can't find neutralizations like this. So, for example, here's a rule from Southern Paiute, from Sapir's grammar. Now Sapir was a pretty careful listener. He talks about the high central unrounded vowel and its various realizations. There are a few things it changes to in the context of č, but one is i. That's tricky, because it's a neutralization. Now, by coincidence, the phoneme i doesn't occur after č, but that's not the point. The point is that the two i's have different sources but are indistinguishable (and Sapir was a careful enough phonetician to distinguish a variety of realizations of i, so I believe him that it's the same). Which means the two are not in complementary distribution – hardly surprising.

Neutralization

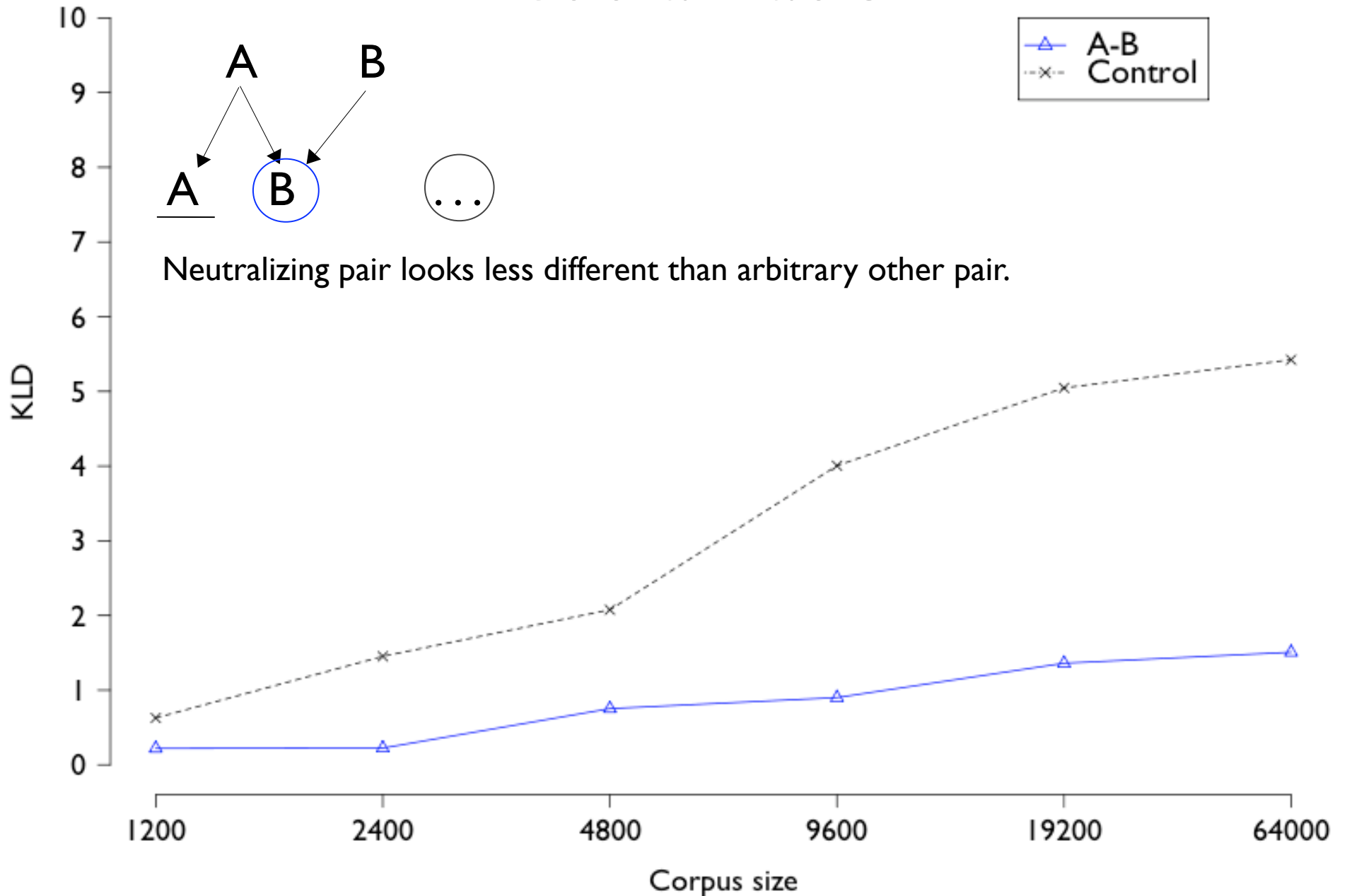
- Artificial language consisting of randomly generated strings from an alphabet of 46 “phonemes”
- One phoneme (A) neutralizes 1 another (B) in exactly eight contexts

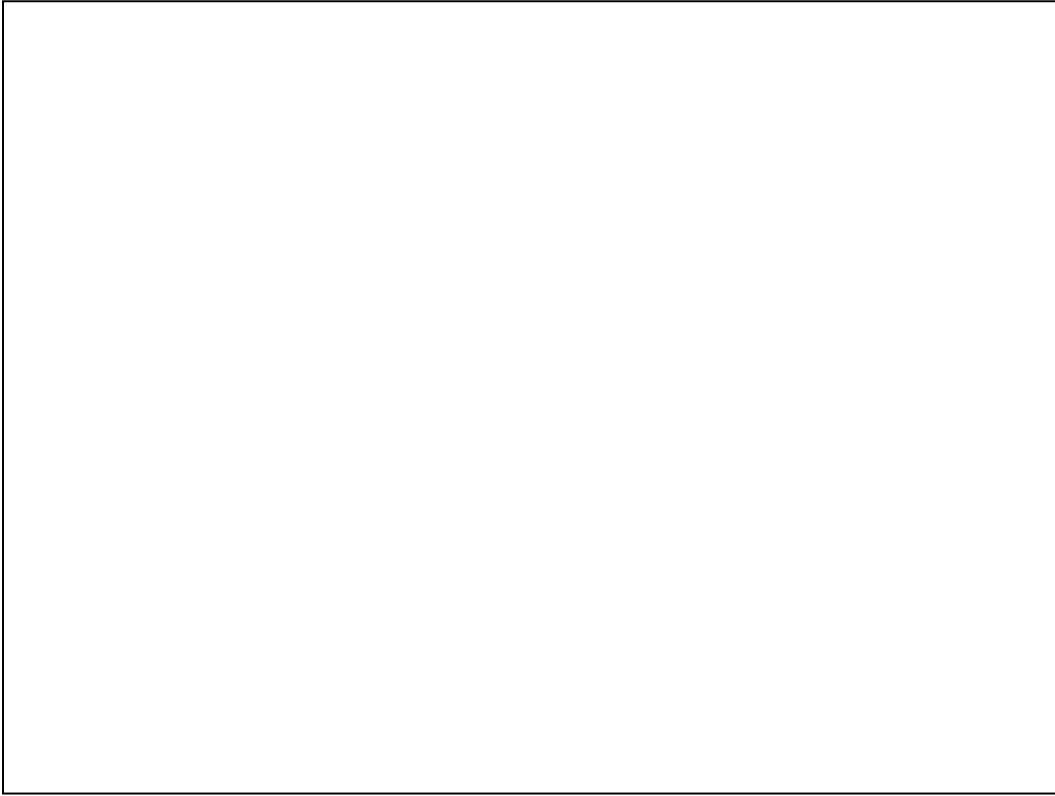




Now let's run this in Peperkamp's framework. Let's do the same experiment they did, but change the single allophonic rule to a single neutralization rule ($A \rightarrow B$, where B is another phoneme).

Neutralization





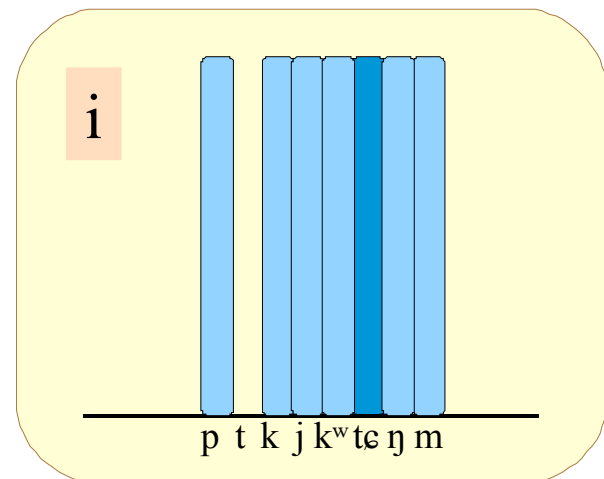
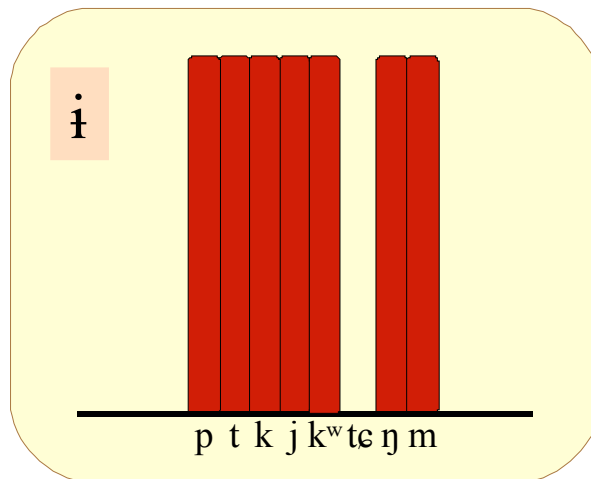
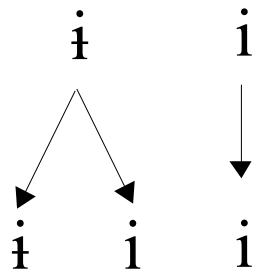
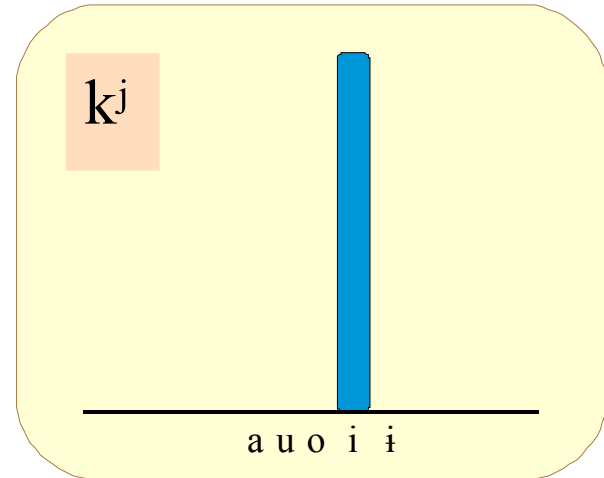
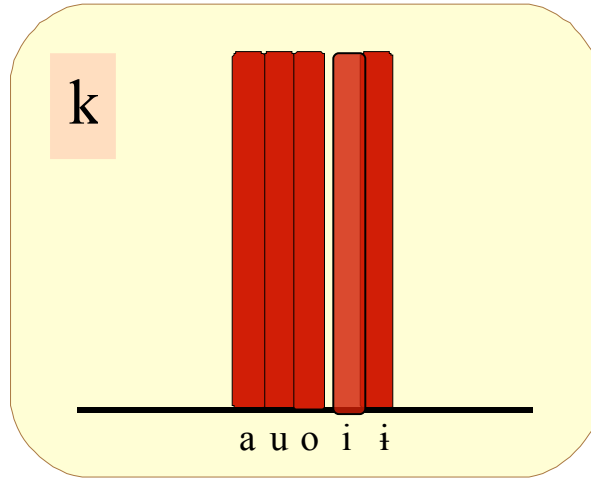
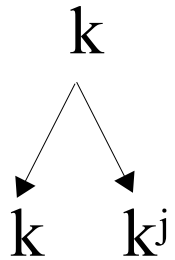
Here we have the Z -scores for A versus B, and for a control; in fact, the relation between the two is *backwards*, so that the related pair has a smaller KLD than the control, which means we have no hope of finding it by this method. But why should we care? This paper was supposed to be about allophony, not neutralization. One reason is simply that this gets you much less far than Peperkamp et al. would make out. . .

Neutralization

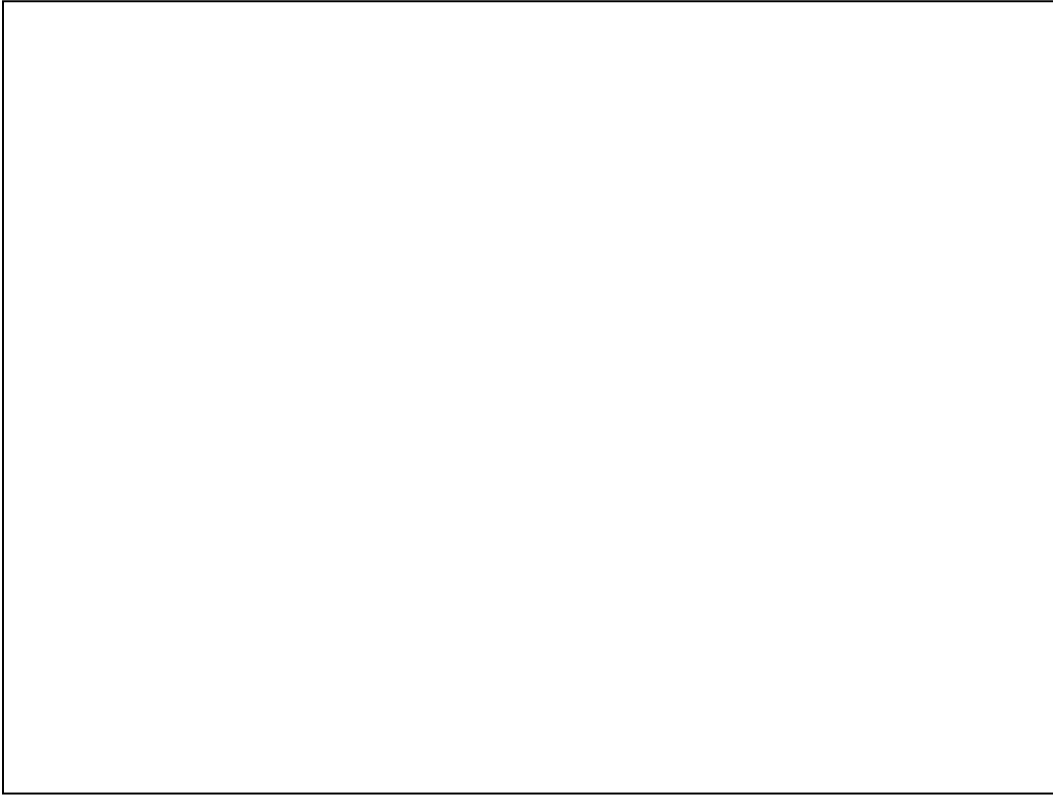
- Actually, this picture is idealized; in particular, it is not a picture of neutralization by itself
- Rather, it is the picture of one neutralization in the presence of another process
- Neutralization by itself *will* have weak KLD evidence in its favour, because the target segment will be slightly more common in certain contexts (unlike a complementary distribution test, in which there will be no evidence for neutralization)
- But the evidence is wiped out by any segment which has a distribution even slightly more unusual than the minor deviation the target shows

Interactions

Counterfeeding



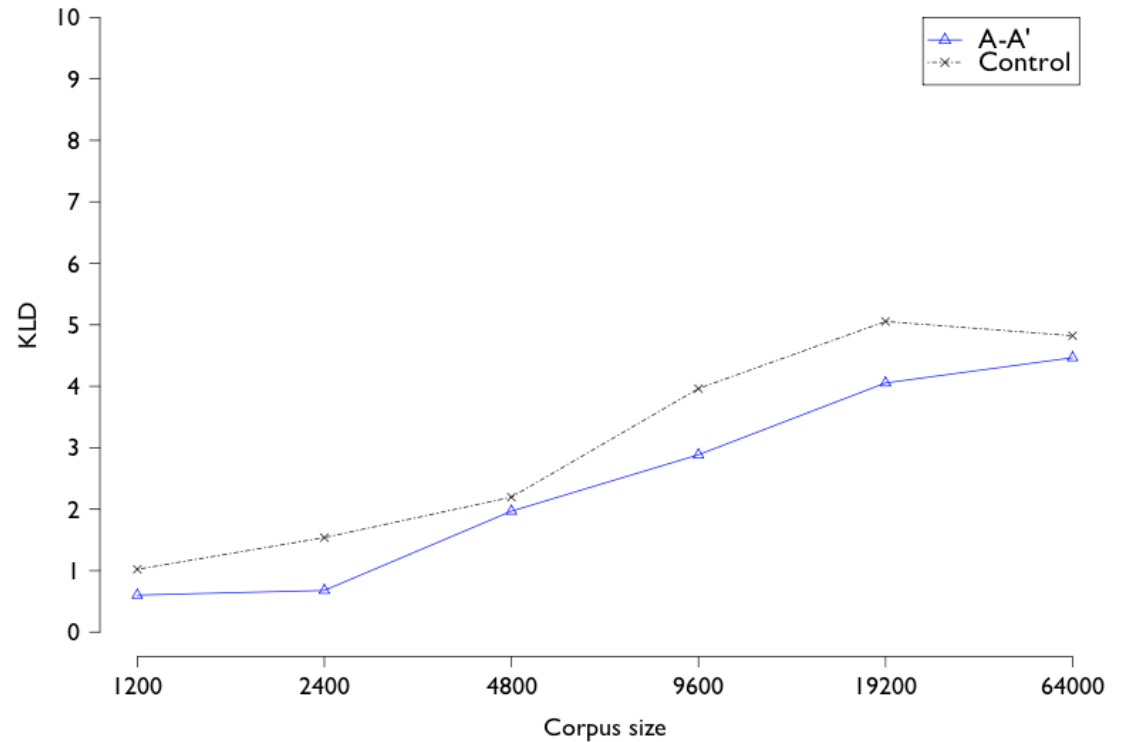
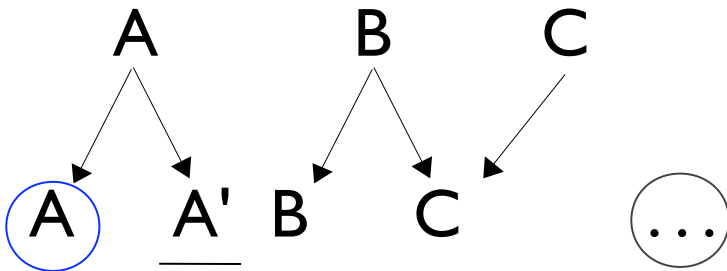
(Southern Paiute; Sapir 1929,
Lovins 1972)

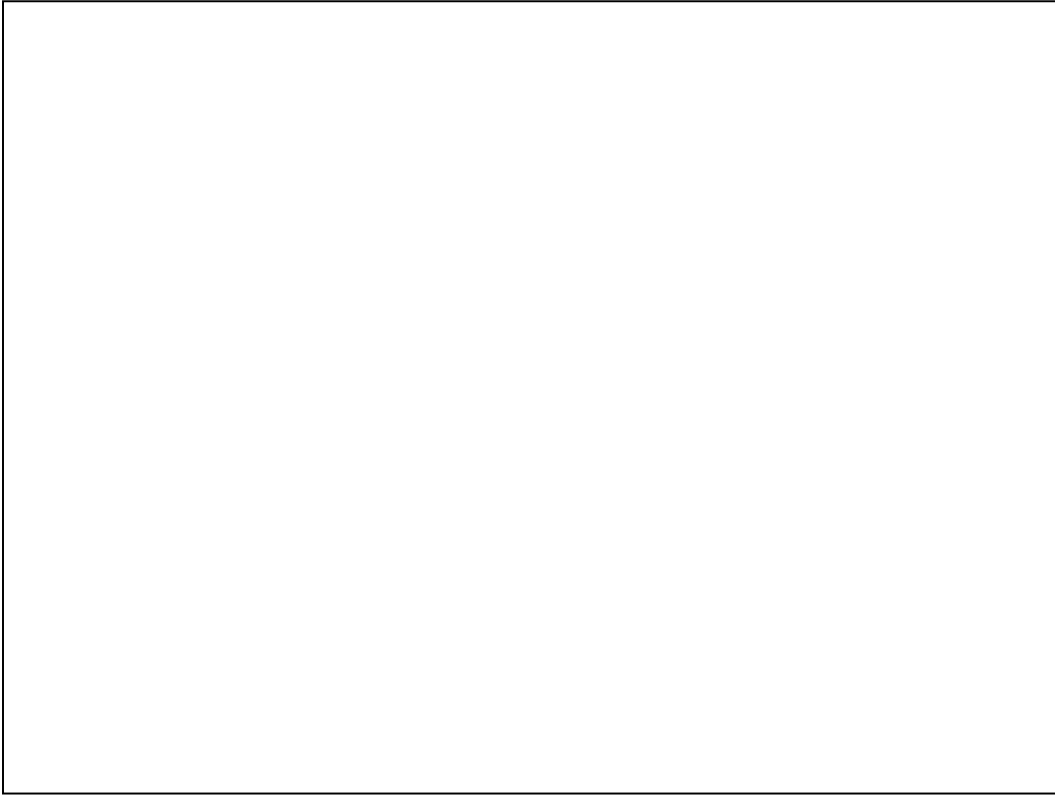


. . . more importantly, neutralization rules can get in the way of allophonic rules. You can imagine Peperkamp's system working iteratively to find multiple allophonic rules (if that could be made to work properly without the problems discussed above), but sometimes the order in which we discover processes needs to be ordered, because the processes themselves are ordered. Here is a classic example of counterfeeding, again from Southern Paiute. Sapir's discussion of the high vowel neutralization concludes with a brief comment that derived *i* does not trigger palatalization the way underlying *i* does. That means that, although the palatalization is an allophonic rule, it doesn't bear the complementary-distribution signature of allophony.

Interactions

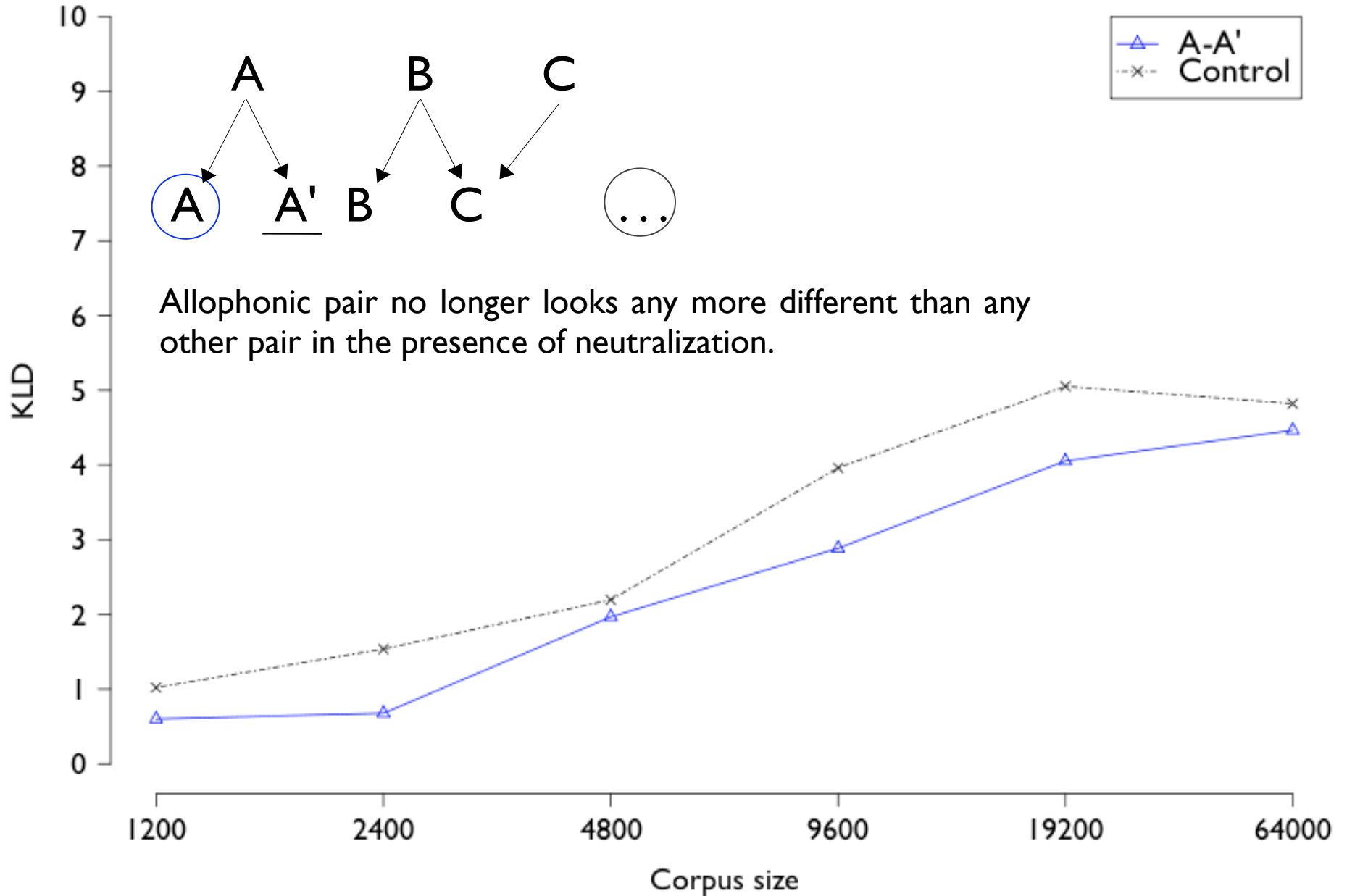
- Artificial language consisting of randomly generated strings from an alphabet of 46 “phonemes”
- One phoneme (A) changes to an allophonic variant (A') in exactly eight contexts
- One of the triggers for the above allophony (B) neutralizes to a non-trigger (C) in exactly eight contexts





Let's do this in Peperkamp's system. Suppose that we take their original experiment, and we add a rule taking one of the triggering contexts for the allophony (B) to another phoneme (C). Now we compare the allophone A' with A and . . .

Interactions





. . . unsurprisingly, the KLD is not distinguishable from control.

KLD Saved?

Peperkamp et al. are not content to stop with toy-language data, but, in developing an artificial French corpus to further test the KLD learner, they note that “spurious allophonic rules might emerge” in case the test mistook low-frequency for non-occurrence. They thus propose a filters on possible allophone pairs.

Phonetic Similarity

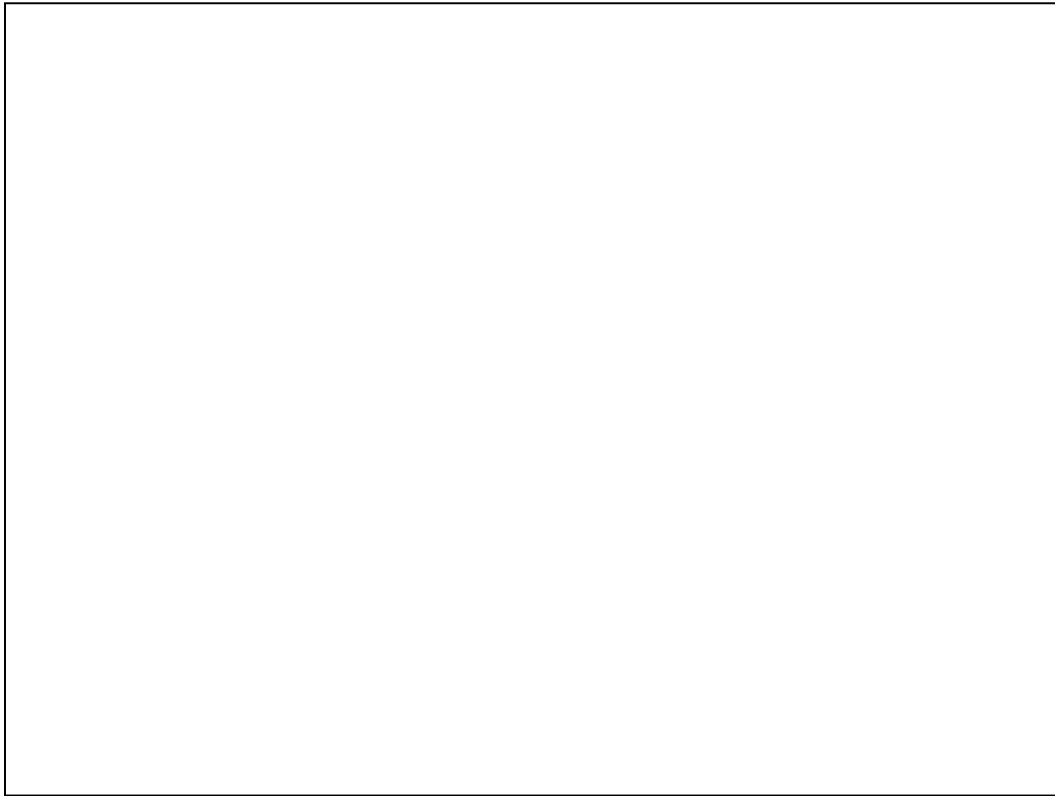
Pick only the most phonetically similar candidate allophone pairs.

$i \rightarrow aj / a[\text{uvular}] _$

(Southern Paiute)

$t \rightarrow \text{ʔ} / V _ V$

(English)



There are thus serious problems with Peperkamp's proposal. It scrapes by on very very simple systems, but is guaranteed not to work on most of phonology. The infant using this system would need to supplement it with a lot. But let's come clean a bit here – the core proposal won't work, but Peperkamp et al. don't just have the core proposal. They attempted a second experiment which they sold as a “natural language” rather than an artificial language test of the system – actually, it isn't really, because they didn't let the rules speak for themselves, but rather, took a phonemic transcription from French CHILDES and implemented their own tightly controlled rules. Nevertheless, they did have *two* rules, on which the system is, according to what we've said here, supposed to fail. Why didn't it? Well, if you read their graphs, it clearly does fail, and so they fix it up. They think it fails for a much more innocuous reason than the real one, however – they point out that any statistical algorithm is prone to what I call “frequency's curse” -- not being sensitive to noise cuts both ways, because you can't tell if what you're hearing is zero frequency plus noise or low frequency. So, they point out, the algorithm might mistake low frequency for nonoccurrence. After all, if something has low frequency, it will give rise to an unusual distribution, which will give a false positive using KLD. That's true, but the problems are much deeper. Their solution to the frequency problem is what gets them out of the multiple independent rule problem, but if they went up against real systems, they would be cooked because of opacity. Furthermore, their appeal to “similarity” to constrain rules and bring them back within the realm of possible phonology seems somewhat half-baked; as is often the case with these things, their definition of “similarity” is arbitrary (they don't even present their feature system up front). Could their similarity metric really tell us that *i* and *ɨ* are just as “similar” as *i* and *aj* in Southern Paiute? They'll need to be. And what about English dialectal glottalization? Do allophonic pairs really need to be maximally similar, or is that simply a tendency (diachronic or otherwise)? We have a generalization of complementary distribution, a system with serious problems, that is going to lead us astray in a number of important ways, and we're trying to patch it up, but not only does that seem to be a rather counterintuitive approach when the problems are so deep, the patch definitely won't do the job.

KLD Saved?

Peperkamp et al. are not content to stop with toy-language data, but, in developing an artificial French corpus to further test the KLD learner, they note that “spurious allophonic rules might emerge” in case the test mistook low-frequency for non-occurrence. They thus propose two filters on possible allophone pairs.

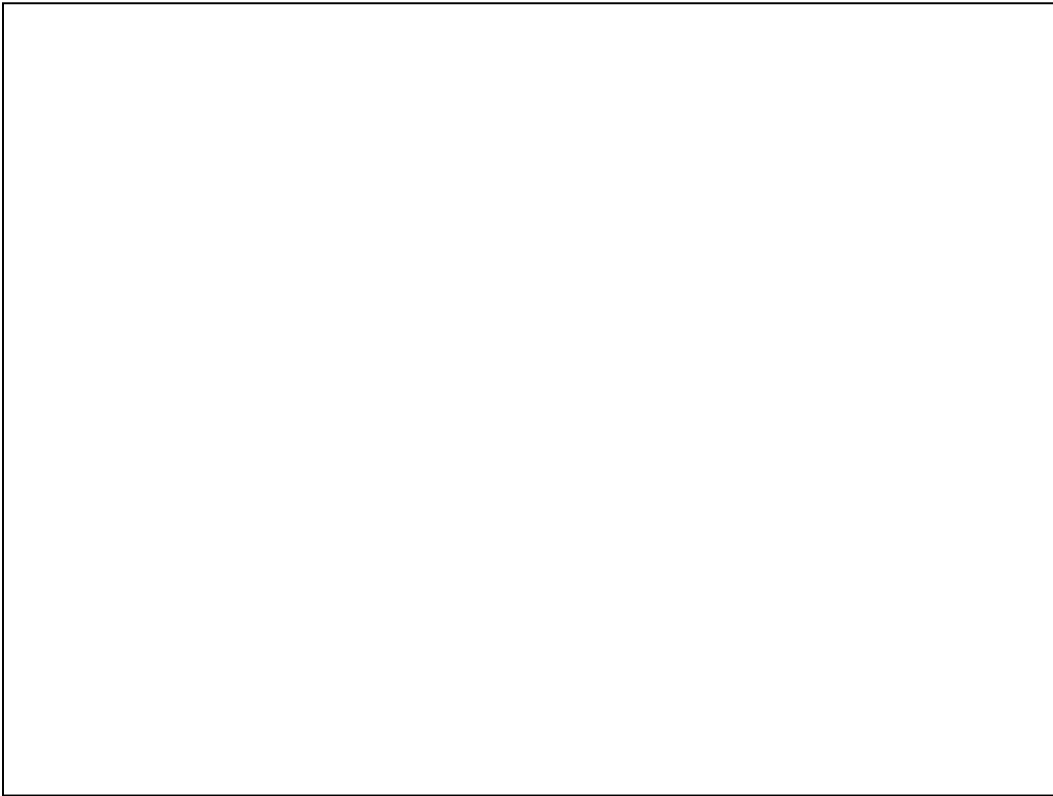
I. Assimilations only

The restricted member of an allophonic pair must have at least one more feature in common with its conditioning environment than the less-restricted member.

$$g \rightarrow x / _ [-\text{high}]$$

(Wuvulu; Blust 2008)

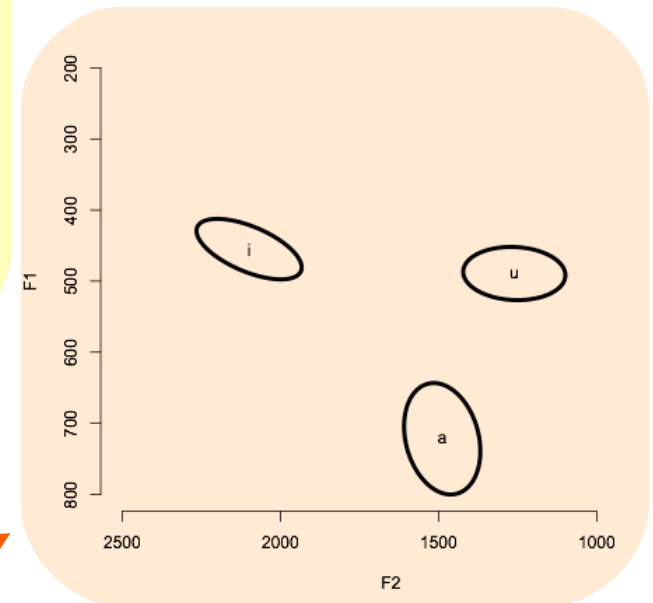
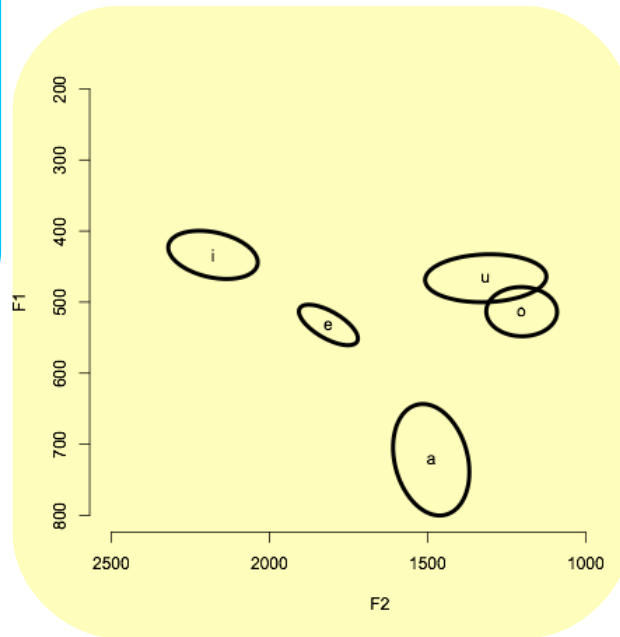
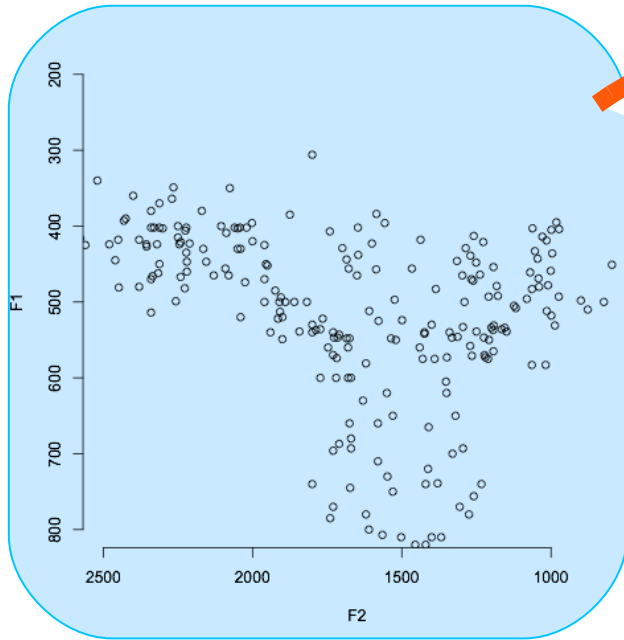
- English vowel length alternation
- Edge effects

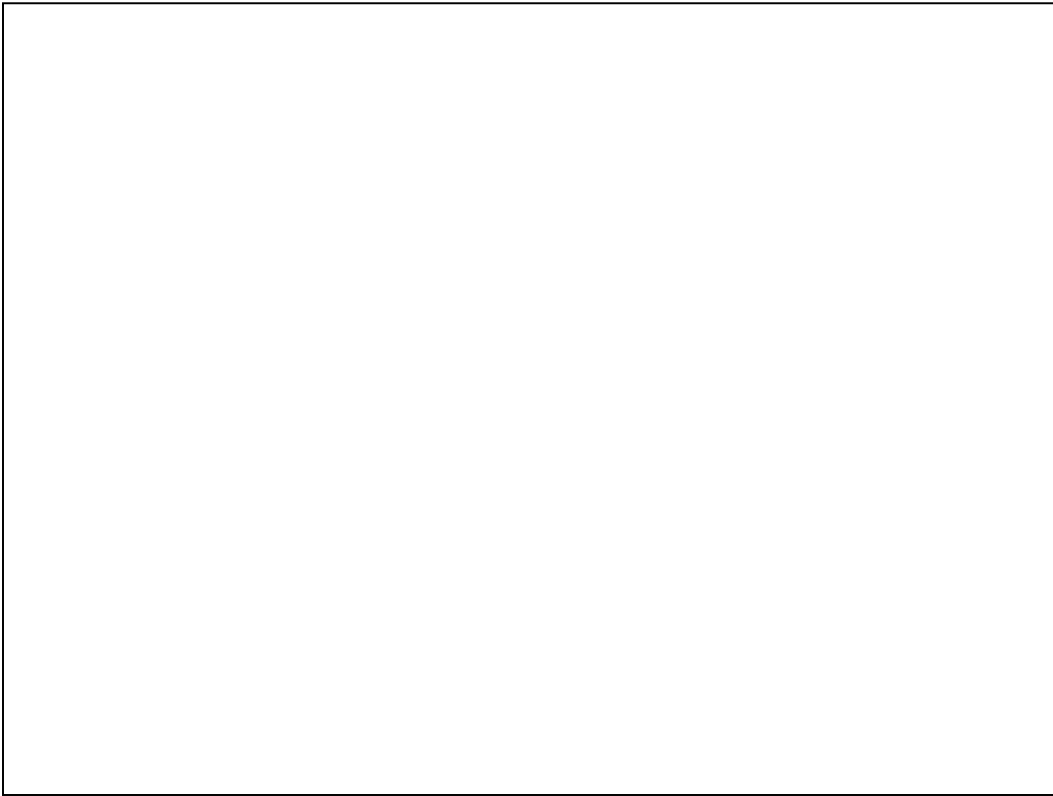


It hardly seems worth mentioning the other constraint. It relies on their secondary system for determining which is the underlying and which the derived member of an allophonic pair, which I haven't discussed, but it does nothing for us. They want to say that we can only get assimilations, so they want to have at least as many features from the conditioning environment on the derived thing as were on it originally (or, actually, at least one more). But if we want to dodge that bullet, all we have to do is turn $A \rightarrow B / C$ around to $B \rightarrow A / C$. In the absence of the phonetic similarity filter, this doesn't get you anywhere.

A Two-State Solution?

Every problem we encountered was a case of putting the wrong two phones together.

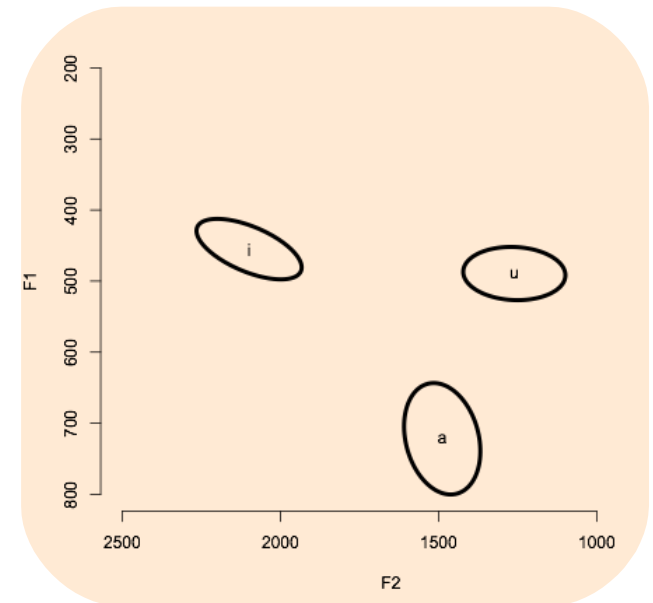
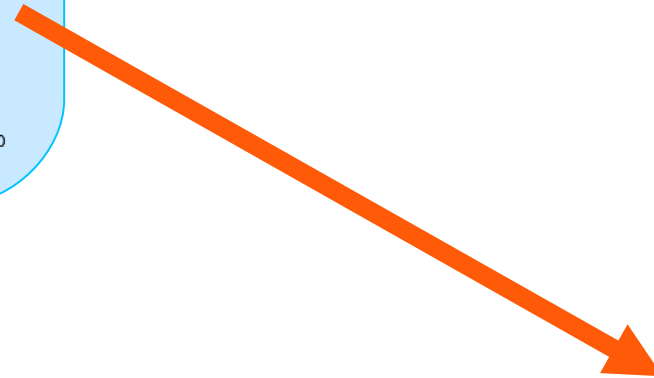
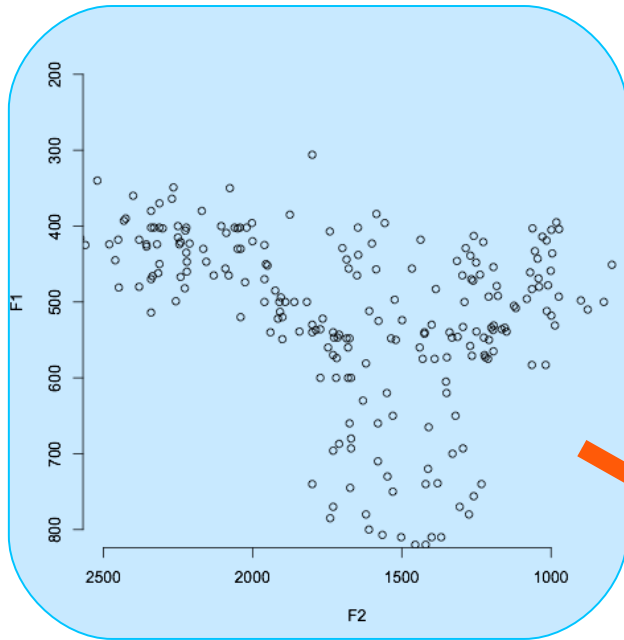




Returning to the bigger picture, the problem here is that we are trying to collapse a large number of small things into a small number of larger things (“agglomerative clustering”). It turns out that's harder than it looks in phonology. It's not a foregone conclusion that this is wrong . . .

A Two-State Solution?

Every problem we encountered was a case of putting the wrong two phones together.





. . . but we shouldn't take it as a foregone conclusion that it's right either.