Two Statistical Approaches to Finding Vowel Harmony

Adam C. Baker
University of Chicago
adamc@uchicago.edu

July 27, 2009

Abstract

The present study examines two methods for learning and modeling vowel harmony from text corpora. The first uses Expectation Maximization with Hidden Markov Models to find the most probable HMM for a training corpus. The second uses pointwise Mutual Information between distant vowels in a Boltzmann distribution, along with the Minimal Description Length principle to find and model vowel harmony. Both methods correctly detect vowel harmony in Finnish and Turkish, and correctly recognize that English and Italian have no vowel harmony. HMMs easily model the transparent neutral vowels in Finnish vowel harmony, but have difficulty modeling secondary rounding harmony in Turkish. The Boltzmann model correctly captures secondary roundness harmony and the opacity of low vowels in roundness harmony in Turkish, but has more trouble capturing the transparency of neutral vowels in Finnish.

1 Introduction: Vowel Harmony and Unsupervised Learning

The problem of child language acquisition has been a driving concern of modern theoretical linguistics since at least the 1950s. But traditional linguistic research has focused on constructing models like the Optimality Theory model of phonology (Prince and Smolensky, 2004). Learning models for those theories like the OT learning theory in Riggle (2004) are sought later to explain how a child could learn a grammar.

A lot of recent research in computational linguistics has turned this trend around. Breakthroughs in artificial intelligence and speech processing in the '80s has lead many phonologists starting from models with well-known learning algorithms and using those models to extract phonological analyses from naturally occurring speech (for an overview and implications, see Seidenberg, 1997). Connectionist models have been used to analyze syllable structure in English, Spanish, Icelandic, and Berber, as well as various stress systems in Larson (1993), and vowel harmony in Hungarian in Hare (1990). Statistical models of phonological learning have a rich history, especially in speech processing (see Rabiner and Juang, 1993; Manning and Schütze, 1999). The goal of this paper is to examines two recent statistical analyses of vowel harmony presented by Goldsmith and Xanthos (2006) and Goldsmith and Riggle (2007).

Vowel harmony interests a wide range of phonologists for a number of reasons. It is widespread among the world's languages, but nowhere near universal. It is a phonotactic constraint that is nonetheless violated by many roots in vowel harmony languages, but it's also an active phonological processes that causes alternations. It interacts with other processes like vowel epenthesis, sometimes in opaque ways. And it is a non-local phenomenon, where harmonizing vowels can often be separated by a large number of consonants or transparent vowels.

The fact that vowel harmony is a violable phonotactic constraint makes it an excellent candidate for statistical analysis. Because the analysis is probabilistic, it can readily capture the regularities we see in vowel harmony systems with no trouble accommodating exceptional forms that occur.

The current study explores methods for the unsupervised learning of vowel harmony from positive, naturally occurring texts. The methods are unsupervised because the learning algorithm is trained on examples that never contain the vowel classes marked. It must instead infer a model that accurately captures the generalizations in the data solely from the surface forms of the words in the language being learned. The data also contains noise; naturally occurring text contains ill-formed structures like acronyms and abbreviations, and no indication is given in the input as to which forms are well formed and which are not. Because the regularities in many vowel harmony languages can be stated without referring to underlying representations and are statistical rather than categorical, vowel harmony patterns can be readily learned from noisy, unlabeled data.

There are three main motivations for doing unsupervised learning from only naturally occurring texts. The first comes from child language learning. Naturally occurring texts approximate the input a child receives from his environment. Both contain noise, and neither contain category or dependency labels to guide the child. One might argue that in phonology, the input to the child is better labeled than it is in naturally occurring texts where each character is treated as a meaningless symbol rather than associated with its phonetic properties. The phonetic properties of the vowels might be useable by the child to put the vowels into natural classes *before* learning the vowel harmony system.

But naturally occurring texts are also in some ways richer than the input the child receives. They come pre-segmented into phonemes. This brings me to the second main motivation for using unsupervised learning. Unsupervised learning methods can be used by phonologists as a tool for inferring a phonology of the language from the readily available language data. This is the primary motivation for the learning algorithms presented by Ellison (1994) in his analysis of Turkish vowel harmony, and it is one of the primary motivations for the present work. A tool that takes a naturally occurring corpus as input and outputs a phonological grammar is useful independent of whether the algorithm it uses corresponds to learning strategies used by children learning the language. The methods studied here not only find vowel harmony when it exists, but also provide quantitative measures of the extent to which the vowels in a language harmonize, which may be useful in deciding to what extent a borderline language such as Asia Minor dialects of Greek have vowel harmony (Revithiadou et al., 2005; van Oostendorp, 2005).

The primary reason for using statistical methods of unsupervised learning from naturally occurring text is to see how far simple models can be put to use for inferring linguistic structure and when more complex, abstract structure is motivated by the data. Typical linguistic theories posit complex, abstract structures. The approach advocated here posits very simple models with minimal abstract structure. I do this, not as an argument that complex, abstract structures are not necessary, but to find out how far simpler structures can take us and when more complicated structures are required. Bayesian statistical analysis, combined with the notion of Algorithmic Complexity from Solomonoff (1997) gives us a formally precise criteria for deciding when abstract structure is justified by the complexity of the data.

1.1 Two methods for analyzing vowel harmony

Goldsmith and Xanthos (2006) present a number of computational methods for separating consonants and vowels in a text and for learning vowel harmony. The most successful method used the Baum-Welch learning algorithm with hidden Markov models (HMMs, see Manning and Schütze, 1999, Chapter 9 for an introduction). HMM learning was shown to correctly classify segments into consonant and vowel classes based on their statistical distributions for English, French, and Finnish. It also correctly separated the vowels of Finnish into front, neutral, and back vowels, accurately modeling the Finnish vowel harmony system.

I will test this method on another vowel harmony language, Turkish, and on two languages without vowel harmony, English and Italian. I show here that the HMM algorithm can be used to find vowel harmony in Turkish even though it is quite different from Finnish vowel harmony, and that it does not spuriously find harmony in English or Italian.

Goldsmith and Riggle (2007) use an information theoretic bigram model to capture the local segment-to-segment phonotactic effects. They augment this simple model with something like an autosegmental vowel tier to model vowel harmony in Finnish. They use a Boltzmann distribution to combine the markedness measures assigned by their segment and vowel tiers into a well-formed probability distribution and compute the description length of the Finnish corpus under both their bigram model and Boltzmann model. They show that incorporating the effect of distant vowels in the vowel tier allows them to shorten their description of the Finnish corpus.

I test the Boltzmann field approach, again on Finnish, Turkish, Italian, and English corpora. I show how this framework captures the vowel harmony facts phonologists have discovered for Finnish and Turkish, and that it fails to find any vowel harmony model for English and Italian. I go on to show that, as expected, vowel harmony languages gain more from adding the autosegmental vowel tier to the model than non-vowel harmonizing languages. I show further that the information theoretic approach reveals generalizations about the phonotactics of Turkish vowels that, as far as I know, have not yet been noted by phonologists.

I will further show how the HMM methods can be put to use in the Boltzmann field information theoretic framework.

1.2 Corpora used

All the corpora used for the experiments reported below can be freely obtained on the internet. The Finnish, Turkish, and Italian corpora were obtained from the Wortschatz collection at Leipzig University, which can be obtained at `http://corpora.uni-leipzig.de/`. For these studies I used word lists made from the 100,000 sentence corpus for each language. The word list was made by first converting each word to lowercase and then discarding any words that contained non-alphabetic characters.

The English corpus I used was the CMU Pronouncing Dictionary, which can be obtained at `http://www.speech.cs.cmu.edu/cgi-bin/cmudict` or accessed through the Python Natural Language Toolkit at `http://nltk.org/`.

1.2.1 Choice of languages

I chose Finnish, Turkish, and Italian because all three languages have plain text corpora available, and the orthographies of all three languages transparently reflect the phonological features relevant to vowel harmony. Finnish and Turkish are important because both are vowel harmony languages, but each has very different properties that should be captured by our models. Using Finnish also allows direct comparison of my results with those of Goldsmith and Xanthos (2006) and Goldsmith and Riggle (2007).

I chose Italian and English because they should turn up a null result. Neither language has vowel harmony. If the learning methods had converged on a result that resembles the results for Turkish or Finnish, it would cast doubt on those methods, so it is important to show that our proposals do not find vowel harmony in English or Italian.

1.2.2 Types or Tokens?

For the experiments I report I used wordlists made from naturally occurring text corpora, rather than the full corpora. In doing so I've assumed that the frequencies of the phonological phenomena in the lexicon are more important than the frequencies of those phenomena in actual speech. The effect of this assumption is that frequent words and infrequent words phonologically equally important. Why make this assumption?

Pierrehumbert (2001) offers some psycholinguistic arguments for the relevance of type frequency phonology. But while the psychological relevance of type frequency is interesting and it makes type frequency worth looking at on its own right, both the HMMs and information theoretic models I'm looking at try to find a model that maximizes the probability of the data they are meant to explain. The data we want to explain are the regularities in the naturally occurring corpus, not the word list constructed from it. This should argue for using token frequency instead.

But the model of the phonology of a language is not meant to stand by itself as a complete analysis of the language. It is only part of the overall grammar. The other parts, such as syntax, morphology, semantics, and the lexicon are intended to be incorporated together in a Minimal Description Length (Rissanen, 1978) analysis of the language corpus. In such an analysis, the phonology of the most common words will only get encoded once in the grammar, which will define its own codeword, shorter than the length assigned by the phonology component, which is used to encode each occurrence of that word in the corpus. de Marken (1996) uses this strategy to learn a lexicon from an unsegmented text corpus. Given this partial representation of the overall grammar of a language, the job of much of the phonology is to optimize the description length of the lexicon, not the corpus. A word list is a closer approximation to the domain of the phonological component of the grammar than the corpus from which that word list was constructed.

1.3 The structure of the paper

Section 2 describes the vowel harmony facts for Finnish and Turkish, and explains exactly which generalizations a model should capture. Section 3 introduces hidden Markov models, explains the details of the trials I've performed, and reports the results. Section 4 gives a detailed explanation of the Boltzmann field model, the trials I performed using that model, and the results of those trials. Section 5 explores the larger theoretical framework emerging from these models and explores what possibilities the framework holds and what work needs to be done to to get there.

2 Finnish and Turkish Vowel Harmony

Finnish and Turkish both have front/back vowel harmony, but the two harmony systems differ in theoretically crucial respects. Finnish has two neutral vowels that do not participate in vowel harmony, so to capture Finnish harmony a model must allow information about preceding vowels pass through intervening neutral vowels. Turkish has secondary roundness harmony for high, but not low vowels, so to capture Turkish harmony a model must allow for two binary distinctions and recognize that some vowels only harmonize along one axis, not both.

2.1 Finnish

Finnish vowels fall into two harmonic classes: front vowels and back vowels. A word can have either all front vowels or all back vowels, but words have a strong tendency not to mix front and back vowels in a single word. The front vowels are ‘y’[y], ‘ö’[ɵ], and ‘ä’[æ]. The back vowels are ‘a’[ɑ], ‘o’[o], and ‘u’[u]. The vowels ‘i’[i] and ‘e’[e] are neutral, and can appear in words with either front or back vowels. This vowel harmony system is summarized in figure 1, from Goldsmith and Riggle (2007).

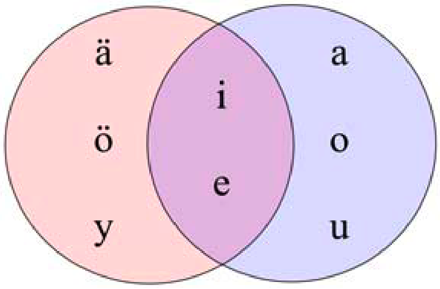


Figure 1: The Finnish vowel harmony system

When a suffix is added to a root, vowels in that suffix will be front or back to match the vowels in the root. This process can even occur when a neutral vowel occurs between a root vowel and the harmonizing suffix vowel. Examples taken from Ringen and Heinämäki (1999) appear in Table 1. Example c shows back harmony through an intervening ‘i’ and example d shows front rounding through an intervening ‘e’.

Ringen and Heinämäki (1999) go on to show that disharmonic roots can cause varying

a.	pöytä-nä	‘table’	essive
b.	pouta-na	‘fine wine’	essive
c.	koti-na	‘home’	essive
d.	käde-llä	’hand’	adessive
e.	kesy-llä	’tame’	adessive
f.	vero-lla	’tax’	adessive

Table 1: Suffix allomorphy from vowel harmony

preferences for suffix vowels. Neither disharmonic roots nor variability in the suffix pose any difficulty for the probabilistic theories used here. These facts will be reflected in the corpus the systems are trained on. The models have no way to distinguish native and borrowed roots, so they will not be able to associate disharmonic stems with foreign borrowings, but they will mark the disharmonic stems as more marked than harmonic stems.

The challenge for a model of Finnish vowel harmony is to correctly identify three classes of vowels: front, back, and neutral. A good model of the Finnish facts will assign lower probability to words containing both strictly front and strictly back vowels, even when a neutral vowel intervenes between disharmonic vowels. It would assign higher probability to words containing only front or only back vowels, again, even if a neutral vowel intervenes.

2.2 Turkish

The vowel harmony system in Turkish raises a different set of challenges for a theory. Turkish has a two-dimensional vowel harmony system. Vowels within words tend to be either all front vowels (ü[y], ö[œ], i[i], e[e]) or all back vowels (u[u], o[o], ı[ɯ], a[a]), and vowels in suffixes are either front or back to match those in the stem.

But there’s also a tendency for the vowels in a word to all be rounded (ü, ö, o, u) or all unrounded (i, e, ı, a). High vowels (i, u, ü, ı) in suffixes are either rounded or unrounded to match the nearest stem vowel, while low vowels (e, ö, o, a) maintain disharmonic lip rounding in a suffix. This vowel harmony system is shown in figure 2. Thus, Turkish vowels are grouped into four harmonic classes, but the vowel system makes use of the full 8 distinctions available to it.

The situation is even more complicated. Kirchner (1993) notes the existence of dishar-

	front	back
rounded	ü, ö	u, o
unrounded	i, e	ı, a

Figure 2: The Turkish vowel harmony system

monic roots and fixed suffixes that do not harmonize with the stems to which they attach. He observes the following generalizations:

1. The vowels [ü,ö,ı] do not occur in roots which are disharmonic with respect to backness, and [ı] does not occur in roots which are disharmonic with respect to rounding.

2. The marked vowels [ü,ö,ı] do not occur in fixed suffix syllables.

So to review, an adequate model of Turkish vowel harmony should meet all the following criteria:

1. Front/back harmony should be more important than round/unround harmony. Front/back disharmony should cause a lower drop in probability than round/unround disharmony.

2. Roundedness disharmony should be costlier (cause a greater drop in probability) in high vowels than in low vowels.

3. Front/back disharmonic occurrences of the vowels [ü,ö,ı] should be costlier than disharmonic occurrences of [a,u,o,i,e].

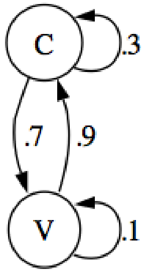
4. Round/unround disharmonic occurrences of [ı] should be costlier than roundedness disharmony in other vowels.

3 Hidden Markov Models

The first vowel harmony modeling technique I'll consider is to estimate the parameters of a hidden Markov model. An HMM is a finite state machine: a finite set of states, a set of arcs between states, and a set of symbols (in this case, the phonemes of the language analyzed). A finite state machine starts in one of its states. It emits a symbol from that state and moves on to another state in the machine, possibly staying in the same state. The machine continues emitting symbols and changing states until it moves into a final state and stops.

An HMM is a probabilistic finite state machine. Each arc out of any state is weighted with a non-negative real number, and all the weights on arcs going out of that state must sum to 1.0. Thus, each state has a probability distribution describing where it will transition to next. Each state also has a probability distribution governing the symbols emitted from that state. Finally, a probability distribution determines which state the model starts in.

Figure 3 shows a example HMM. This machine is defined over the alphabet {'a','b'}; it assigns probabilities to strings of 'a's and 'b's[1]. We can calculate the probability of taking a particular path through the machine and seeing a particular string by multiplying the relevant state transition and emission probabilities. For example, if we assume that the start probability for each state is 0.5, then the probability of seeing the string 'ba' with the state sequence 'CV' is calculated as in equation 1. We can find the probability the model assigns to the string by summing over all the possible paths the model could have taken to generate the string. The probability of the string 'ba' is shown in equation 2.



	C	V
a	0.4	0.8
b	0.6	0.2

Figure 3: A toy HMM: The states and transitions (left) and state emission probabilities (right).

$$\begin{aligned} p(CV, \text{`}ba') &= p(start = C)p(\text{`}b'|C)p(V|C)p(\text{`}a'|V) \\ &= 0.5(0.6)(0.7)(0.8) \\ &= 0.168 \end{aligned} \tag{1}$$

[1] Technically, it assigns a probability distribution to all the strings of 'a's and 'b's of the same length. In order to get a probability distribution over all the strings, we must define a probability distribution over string lengths. Then the probability of the string is the probability assigned by the HMM times the length probability.

$$\begin{aligned} p(\text{`}ba') &= \sum_X p(X, \text{`}ba') \\ &= p(CV, \text{`}ba') + p(CC, \text{`}ba') + p(VC, \text{`}ba') + p(VV, \text{`}ba') \\ &= 0.168 + 0.036 + 0.027 + 0.008 \\ &= 0.239 \end{aligned} \tag{2}$$

HMMs can be used to discover categories in natural language. By setting up an HMM and reapportioning the transition and emission probabilities so as to maximize the probability the HMM assigns to the corpus data, we can see which states prefer to emit which symbols, and call all the symbols that prefer one state to all the others a natural class. Here I only consider complete finite state machines, where every state can transition to every other state, and every state is capable of emitting any symbol from the alphabet. In other words, I start learning with models that assume no zero probability state transitions or emissions.

There is a well known algorithm for estimating the transition and emission probabilities of an HMM to maximize the probability that HMM assigns to the corpus. It is the Baum-Welch algorithm, described in Chapter 9 of Manning and Schütze (1999), and described in greater technical detail in Rabiner and Juang (1993). The intuition behind the algorithm is simple. Pick a starting set of parameters for your HMM. Then run the corpus through the HMM using those parameters and count how many times each state transition happens and how many times each character is emitted from each state. You then normalize those counts to obtain new probability distributions for your transition and emission parameters. This process can repeated over and over, and each time the new parameter settings are guaranteed to improve the probability that the HMM assigns to the corpus.

The Baum-Welch algorithm finds local maxima, not global maxima, so starting with different initial parameter settings can lead to different results, as we will see when considering Turkish and Finnish vowel harmony.

For most of my tests the HMMs had only two states. With a two state HMM, there are two important types of models one could end up with after training it on corpus data, alternating models and harmonic models. In alternating models both states have a high probability of changing to the other state, while in harmonic models both states have a high probability of staying in the same state. This schema is illustrated in Figure 4. The general transition possibilities is shown on the left. If α and β are both large, the machin is an alternating machine, like the one in the center. If α and β are small, the machine is harmonic, like the one on the right.

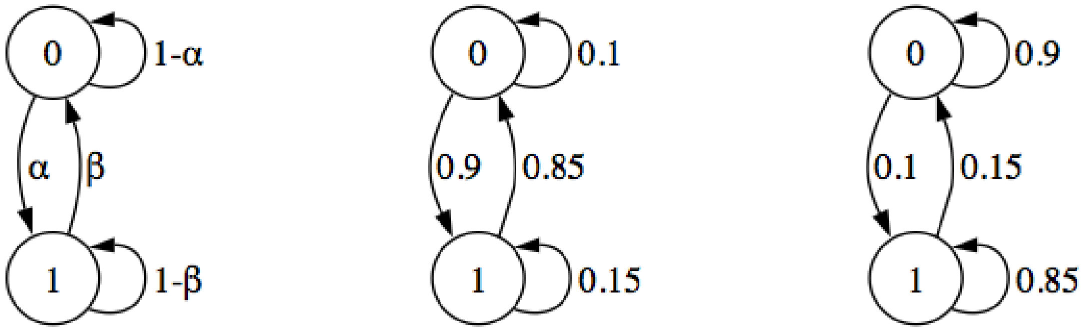


Figure 4: General HMM schema (left), alternating HMM (center), harmonic HMM (right).

A language can be tested to see if it has vowel harmony, and if so, what letters belong together in the same vowel classes, by training an HMM on a corpus with all the consonants removed from it. Vowel harmony languages should end up with harmonic state transitions and with the each harmonic class of vowels preferring to be emitted from the same state. Neutral vowels will show only a very weak preference for one state or the other. Languages without vowel harmony may end up with alternating transitions, or with some other non-harmonic transition parameters.

One might wonder if we have a good basis for filtering the consonants out of the language. Gross phonetic differences between consonants and vowels may justify grouping them in different natural classes, but their statistical distribution can also be used to separate them as well. I turn to this before returning to the case of vowel harmony.

3.1 Discovering consonants and vowels

To learn the vowel/consonant distinction, all we have to do is run our training algorithm with a two state HMM on the corpus. The result will be an alternating model, which the vowels being emitted from one state and the consonants being emitted from the other. This works because generally, consonants and vowels alternate in natural language.

The HMMs started training with transition probabilities very near 0.5 for all state transitions, shifted a small bit away from 0.5. The emission probabilities for both states were set to the frequency of the letter in the corpus. The start probabilities were assigned values of 0.5 for each state before each re-estimation of the parameters to prevent the HMM from converging towards a model that just used one state to predict the first character of the word.

The data in this section replicate the findings of Goldsmith and Xanthos (2006) for English and Finnish, and extend the results to Turkish and Italian.

3.1.1 English and Finnish

My results using the HMM learning algorithm on word lists from both English and Finnish replicate Goldsmith and Xanthos (2006). The found perfect separation of English Cs and Vs in the two states, and near perfect separation of Cs and Vs for Finnish, only miscategorizing Finnish "q".

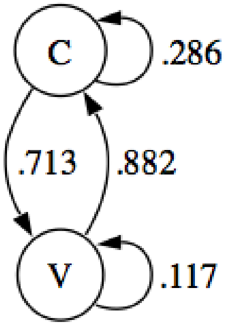
My tests used different starting parameters but arrived at similar results: perfect separation of vowels and consonants for both English and Finnish. My test even sorted "q" correctly with the consonants in Finnish, though it only showed a very slight preference for the consonant state over the vowel state.

3.1.2 Italian and Turkish

The same experiments performed on both Turkish and Italian corpora also separated the vowels from the consonants perfectly, with both results in alternating models. Figure 5 shows the results for the Italian corpus. Figure 6 shows the results for the Turkish corpus.

For each, the graph on the right shows the two states, labeled C and V *post hoc*, with the transition probabilities labeled on the arcs. For instance, Figure 5 shows the V state for the HMM trained on Italian has an 11% chance of staying in the V state and an 88% chance of transitioning to the C state.

The table on the left of each figure shows the log ratio of the emission probabilities for each character in the language's alphabet. If O is an emitted character, P is a phoneme and S is a state, then the log ratio shown in the tables for P is $log_2(pr(O = P|S = V)/pr(O = P|S = C))$. The tables show that for both Italian and Turkish, the consonants strongly preferred the C state and the vowels strongly preferred the V state. The smallest preference is the Italian character ‘p’, with a log ratio of -3.2. This means that ‘p’ is still more than 8 times more likely to be observed from the C state than from the V state.



C-state		V-state	
phone	log ratio	phone	log ratio
c:	-11.70	a:	161.48
b:	-51.25	e:	276.89
d:	-239.8	i:	50.425
g:	-51.06	o:	150.44
f:	-27.21	u:	57.976
h:	-64.50		
m:	-367.6		
l:	-284.9		
n:	-312.7		
q:	-343.1		
p:	-3.204		
s:	-37.58		
r:	-324.5		
t:	-45.19		
v:	-470.0		
z:	-267.0		

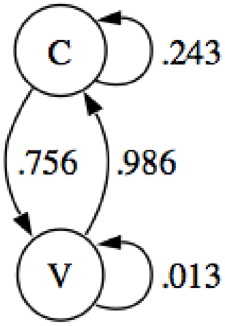
Figure 5: Transition probabilities (left) and emission preferences (right) for Italian two state HMM

3.2 Discovering Vowel Harmony

Having shown that we can accurately pick the consonants from the vowels in a corpus, I now turn the HMM learning algorithm on the same corpora with the consonants removed.

3.2.1 Finnish Vowel Harmony

We would expect a two-state HMM that has successfully learned this system of vowel harmony to separate the front vowels into one state and the back vowels into the other, with each state having a very high probability of staying in itself and a very low probability of



C-state		V-state	
phone	log ratio	phone	log ratio
ğ:	-829.7	ö:	593.5
ç:	-205.1	a:	12.45
c:	-259.6	e:	92.29
b:	-481.5	i:	11.90
d:	-582.3	o:	7.407
g:	-233.8	u:	11.95
f:	-227.9	ı:	13.44
h:	-224.4	ü:	349.7
k:	-328.0		
j:	-557.8		
m:	-665.8		
l:	-903.9		
n:	-981.6		
q:	-5.424		
p:	-4.767		
s:	-168.6		
r:	-993.6		
t:	-6.885		
w:	-49.68		
v:	-641.6		
y:	-498.1		
x:	-4.029		
z:	-842.6		

Figure 6: Transition probabilities (left) and emission preferences (right) for Turkish two state HMM

transitioning to the the other state. We would also expect the neutral vowels to show a very weak preference for one state or the other, or no preference at all.

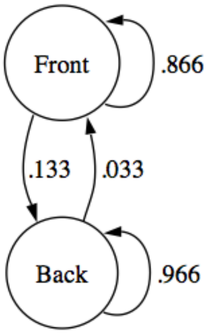
Starting with emission probabilities equal to the observed frequencies of each vowel, the final model is not what we expected. Instead, the resulting model had one state with an 86% chance of remaining in itself and only a 14% chance of transitioning to the other state. This state was preferred by [i,e,ä,ö]. The other state transitioned to itself only 20% of the time and back to the first state 80% of the time, and was preferred by [a,o,u,y]. The machine did not fall into a harmonic configuration, nor did it separate the vowels into the correct harmonic classes.

The HMM failed to correctly infer the Finnish vowel harmony system because it started too close to a local probability maximum. Starting from a different set of starting parameters could lead to different ending parameters. Instead of setting the emission probabilities for each vowel in both states to that vowel's empirical frequency, I assigned a random emission probability to each vowel, state pair. The result was a starting parameter setting where each vowel had a random bias towards being emitted from one state or the other, whereas before each vowel had no bias towards one state or another.

Again, we predict that one state to be preferred by [ä,ö,y] and the other to be preferred by [a,o,u], with [i,e] sitting the fence between the two states. We expect the two states to both prefer to transition back into themselves rather than transition away to the other state. Figure 7 shows that this is exactly what we get. These results replicate those reported by Goldsmith and Xanthos (2006).

Five random trials were run on the Finnish corpus, and each trial converged to the model in Figure 7. This suggests that the local maximum found in my first trial covers only a small part of the total space of possible starting parameters. Most of the space of possible starting parameters seem to converge on the model in Figure 7. I computed the total probability assigned to the corpus by both of the final models, and the correct model in Figure 7 is indeed the better model.

This model also adequately captures all of the facts we want noted in section 2.1. Disharmonic vowels require the machine to either make a very low probability transmission or a low probability emission, so words with disharmonic vowels will be marked, that is, less probable. Neutral vowels are emitted by either state with nearly equal probability, so the best parse for an word will assign the neutral vowels to the same class as the other vowels in the word. This means that disharmonic vowels are still penalized, even when a neutral



Front-state		Back-state	
phone	log ratio	phone	log ratio
e:	0.741	a:	-8.436
i:	0.180	o:	-2.794
ö:	335.9	u:	-18.74
y:	136.7		
ä:	319.9		

Figure 7: The final transition and emission parameters for an HMM trained on a corpus of Finnish vowels

vowel intervenes, because it still forces a costly state transition that could have otherwise been avoided.

3.2.2 English and Italian

What happens if we apply the same technique that we used to discover Finnish vowel harmony on to a text corpus from non-vowel harmony language? Will we still discover vowel harmony? Will we get an alternating model instead? Or something else perhaps? To find out, I trained two state HMMs on English and Italian corpora of only vowels. I chose the starting parameters the same way I did for the V/C tests.

The HMM learning algorithm did not find any vowel harmony in either English or Italian. The English results are shown in figure 8. The Italian results are shown in figure 9.

The English results show neither harmony nor alternation, but instead form a sink, with state 0 staying in state 0 and state 1 changing to state 0. No vowels show any strong preference for state 1. These results are difficult to interpret or draw any meaningful generalization from.

The Italian results did separate the vowels into two natural classes. One state preferred to emit the back vowels (a, u, o) and the other preferred to emit the front vowels (i, e). But as figure 9 shows, these preferences typically weren't very strong, frequently less than 2:1. The final parameters also ended in an alternating configuration, although again the tendency was not very strong. The back vowel state only had a 55% probability of transitioning to the front vowel state, leaving a 45% likelihood of staying in the back vowel state. These findings may show a slight tendency for Italian vowels to alternate between front and back, but the tendency is only slight.

Since starting with the empirical frequencies of each vowel as the emission parameters caused the Finnish HMM to converge to a sub-optimal solution, we should worry that the same is true for English and Italian as well. I ran five trials with random emission parameters for each language, and all trials converged to the same models reported in section 3.2.2. The HMM did not find vowel harmony in English or Italian.

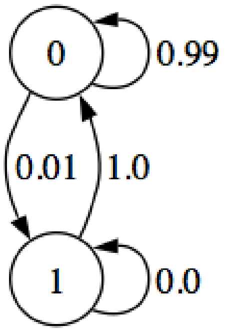
3.2.3 Turkish

Recall that Turkish has primary front/back harmony and secondary roundness harmony in high vowels. From this one would predict that a two state HMM would most likely be harmonic, separating the vowels into front and back. One might not be surprised if instead the HMM separated the vowels into rounded and unrounded vowels. As for Finnish, starting with emission probabilities equal to empirical frequencies produces unexpected results, shown in figure 10.

The final HMM parameters grouped e, i, and ı together, and nearly placed a in that group as well. The model almost separated the vowels into rounded and unrounded classes. The transition probabilities also lead to a harmony model, although the round state only has a slight preference for staying in the round state rather than changing state.

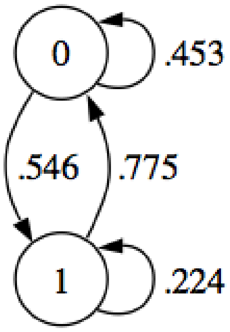
Again starting with random emission parameters leads to convergence on a model that reflects our expectations by capturing front/back harmony. These results are shown in Figure 11.

I ran five random trials, and four of the five random trials converged on the model in Figure 11, while only one of the five random trials converged on the model in Figure 10. It's likely that the maximum found in Figure 11 will be reached by the majority of possible starting configurations, with the maximum in Figure 10 is only reachable from a small portion



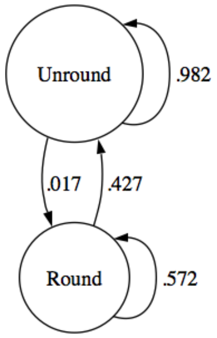
State 0		State 1	
phone	log ratio	phone	log ratio
IY	101.8	AA	-1.81
AH	199.2	AE	-2.69
OW	0.251	EH	-1.59
ER	296.8	AW	-1.77
		UW	-1.57
		AO	-1.73
		IH	-0.12
		AY	-1.28
		EY	-0.02
		UH	-1.62
		OY	-1.24

Figure 8: Transition probabilities and emission preferences for an HMM trained on an English vowel corpus



State 0		State 1	
phone	log ratio	phone	log ratio
a:	-2.590	i:	7.307
u:	-0.794	e:	0.551
o:	-0.552		

Figure 9: Transition probabilities and emission preferences for an HMM trained on an Italian vowel corpus



Round-state		Unround-state	
phone	log ratio	phone	log ratio
a:	-0.140	e:	1.794
o:	-3.591	i:	3.441
u:	-171.6	ı:	45.15
ö:	-8.487		
ü:	-107.2		

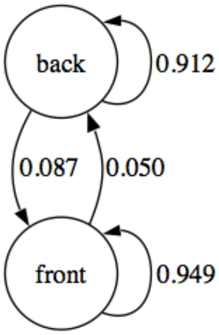
Figure 10: Final transition and emission parameters with a Turkish vowel corpus, empirical starting emissions.

of the space of initial parameters. Computing the probability of the corpus under the two resulting models confirms that the model in Figure 11 with the front and back states is a better model than the one in Figure 10 with round and unround states.

The model in Figure 11 captures some of the generalizations noted in section 2.2. It succeeded at capturing front/back harmony as the most statistically important vowel to vowel effect. It also captures the fact that disharmonic occurrences of the cross-linguistically rarer vowels [ö,ü,ı] are more heavily penalized than disharmonic occurrences of the other vowels. ö and ı have an overwhelming preference for the front and back state, respectively, so emitting them from the other state will result in a massive drop in probability compared to an emission of one of the more common vowels from its dispreferred state. ü, while not so strongly tied to the front state, still prefers the front state more strongly than any of the unmarked vowels prefer their states, and thus incurs a greater penalty when it occurs in a disharmonic word.

A two state HMM has no way of capturing both front/back and roundness harmony, though, nor can it capture the fact that roundness harmony occurs with high vowels and not low vowels. But since Turkish has two harmony axes, defining four harmony classes rather than two, a four state HMM should better capture Turkish vowel harmony.

I ran the same HMM experiments with a four state HMM using both an empirically determined, unbiased starting emission probabilities and with random starting emission prob-



back		front	
phone	log ratio	phone	log ratio
a:	2.886	e:	-5.792
o:	1.293	i:	-3.025
u:	5.565	ö:	-357.4
ı:	529.1	ü:	-7.599

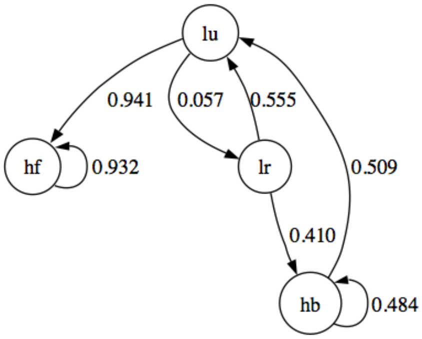
Figure 11: The final transition and emission parameters for an HMM with random initial emission probabilities

abilities. The results are shown in figures 12 and 13. Rather than showing all the transition arcs, I only show transition arcs with probability greater than 0.05. The preference numbers displayed in the tables are the log ratios of the emission probability of the character from it preferred state over the total emission probability of the character from all other states. It measures how strongly that character prefers to be emitted from the state with the highest probability of emitting that character as opposed to any other state. Negative values mean that the total emission probability for that character from the other three states was greater than the emission probability from that charcter's preferred state.

The first experiment with the emission probabilities set to the empirical frequency failed to separate the vowels into the front/rounded, front/unrounded, back/rounded, back/unrounded classes we hoped it would. Instead, it separated into high and low vowels, with high vowels separated by frontness or backness and low vowels separated by roundedness.

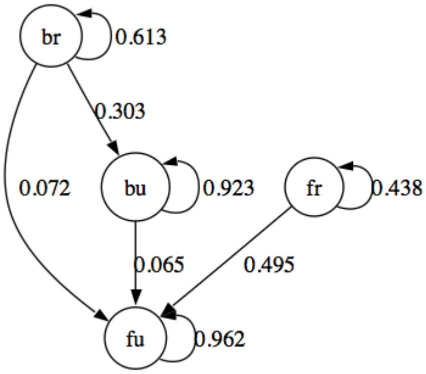
Randomizing the initial emission probabilities causes the HMM to sort the vowels into the four desired classes. The states end up being mostly harmonic, with the most probable transitions away from any state landing in the other state with the same front/back bias. This is the results we expect, since front/back harmony is primary in Turkish.

The model in Figure 13 also captures the fact that round/unround harmony is stronger for high vowels than it is for low vowels by assigning the high vowel for each state a much stronger preference for that state than its low counterpart. It also captures the fact that



low unround		low round		high unround		high round	
phone	pref	phone	pref	phone	pref	phone	pref
a:	-0.04	o:	1.15	i:	0.99	u:	999
e:	-0.03	ö:	4.94	ı:	4.78	ü:	10.07

Figure 12: The final parameter settings for a four state HMM trained on Turkish vowels



front unround		front round		back unround		back round	
phone	pref	phone	pref	phone	pref	phone	pref
i:	1.65	ü:	999	ı:	6.71	u:	7.01
e:	0.60	ö:	12.55	a:	0.34	o:	1.59

Figure 13: The final parameter settings for a randomly intitialized four state HMM with trained on a Turkish vowel corpus

the front round vowels [ü,ö] are more marked in disharmonic words by giving them a much stronger preference for the front round state any other vowel's preference for it's associated state. It doesn't do as well at associating [ı] with the back unround state, but it still gives ı a pretty strong preference for the back unround state. The model in Figure 13 does a good job of capturing the generalizations observed by Kirchner (1993).

3.3 Conclusions

The findings presented in this section reveal a very successful algorithm for finding vowel harmony from a corpus sampled from a given language.

1. Separate the vowels and consonants of the language using with a two state HMM trained on a word list constructed from the corpus (Section 3.1).

2. Eliminate the consonants from the word list and train a two state HMM on this new word list, randomizing the initial emission parameters.

3. Repeat step 2 a few times.

4. Choose the model generated in step 2 that assigns the best probability to the vowel word list. If both states in this model have a high tendency not to change to the other state (more than 70%), conclude that the language has vowel harmony. The vowels that all prefer to be emitted by one state over the other form a harmonic class.

This algorithm is very successful at discovering vowel harmony in Finnish and rejecting Italian and English as vowel harmony languages. It was less successful at capturing Turkish's more complicated vowel harmony system, but by using four states it was able to learn the generalizations phonologists have observed about the Turkish vowel harmony system.

The Turkish trials also reveal an interesting question when using HMMs to categorize symbols: how many categories should we look for? When should we use two state HMMs, and when should we try four, or some other number of states, when attempting to analyze a new language whose vowel properties are unknown to us?

Nonetheless, I have shown that HMMs can be used to discover vowel harmony in different vowel harmony systems when there is vowel harmony, and that they do not spuriously find vowel harmony in languages that do not have vowel harmony.

4 Phonological Forces in a Boltzmann Field

The next model of vowel harmony I consider uses simple bigram models of distant vowel interactions. The goal of this model is to minimize the number of bits needed to encode a corpus by using the statistics of phoneme distribution in the lexicon to predict the identity of a segment based on the surrounding segments. The number of bits needed to encode a character is roughly the number of yes/no questions needed to identify that character with certainty.

Because vowel harmony reduces the number of possible vowels that can occur in a word once the harmony class has been chosen, it will reduce the number of questions needed to correctly guess the identity of a vowel (given knowledge of another harmonic vowel in the word). So vowel harmony should be included in a good model of a vowel harmony language in this framework.

4.1 Background: Bayes' Theorem and Minimal Description Length

But why should we care about how many bits it takes to encode a corpus? If we assume the goal of a scientific investigation is to find the most likely hypothesis given the observed data, then finding this most likely hypothesis is equivalent to finding the most succinct description of the hypothesis and the data. This follows as a consequence of Bayes' Theorem and Shannon's Information Theory (Shannon and Weaver, 1949).

Bayes theorem is stated in (3).

$$p(A|B) = \frac{p(B|A)p(A)}{p(B)}. \tag{3}$$

Let g be a grammar, our hypothesis about the language, selected from a space of possible grammars G. Let c be a corpus, the data our grammar is supposed to explain. Then it follows from Bayes Theorem that the probability of the grammar g given the corpus c is:

$$p(g|c) = \frac{p(c|g)p(g)}{p(c)} \tag{4}$$

The meaning of the $p(c|g)$ factor is clear. If g is a probabilistic grammar, then $p(c|g)$ is the probability that g assigns to the corpus c, written $g(c)$. It's less clear what to make of the $p(g)$ factor, but see Section 5 for some discussion of this and its importance. It's also not clear what $p(c)$ could be, but it doesn't matter. We don't need to find $p(g|c)$, we just need to find the grammar g that maximizes that quantity. But $p(c)$ is constant as we vary g, so it can be safely ignored.

$$\operatorname*{argmax}_{g\in G}(p(g|c)) = \operatorname*{argmax}_{g\in G}\left(\frac{p(c|g)p(g)}{p(c)}\right) = \operatorname*{argmax}_{g\in G}(g(c)p(g)) \tag{5}$$

For any event A the information of A is defined to be $-log_2(p(A))$. Goldsmith and Riggle call this the $plog(A)$, for positive log probability (recall that the log of a number between 0 and 1, such as a probability, is negative, so multiplying by -1 gives a positive number). Taking the plog of (5) gives (6).

$$\operatorname*{argmax}_{g\in G}(p(g|c)) = \operatorname*{argmin}_{g\in G}(-log_2(p(g|c))) = \operatorname*{argmin}_{g\in G}(plog(g(c)) + plog(g)) \tag{6}$$

Shannon proved that the plog of an event is a lower bound on the number of bits needed to encode that event in a prefix-free encoding of the event space. He also showed that encoding schemes can come arbitrarily close to this lower bound. This means that $plog(g(c))$ is the number of bits required to encode the corpus using the grammar g, and assuming a prefix-free encoding of all the grammars in the grammar space G, then $plog(g)$ is the length of the grammar (in bits) under that encoding.

To sum up, finding the most probable grammar given a corpus of data is equivalent to the grammar such that the length of the corpus encoded with that grammar plus the length of that grammar is smaller than any other grammar one could choose. This is the principle of Minimal Description Length, which has been put to use for linguistic analysis by, for example, Ellison (1994), de Marken (1996), and Goldsmith (2001).

For a more detailed introduction to these concepts and their applicability to linguistics, see Goldsmith (2007a).

4.1.1 Unigram and bigram models

The unigram and bigram grammars are very simple language models that form a good baseline to start from. A unigram model assigns each segment a probability equal to its empirical frequency in the corpus. The total probability of the corpus is the product of the probability of all the individual segments that occur in it. A bigram model assigns probabilities to two letter sequences based on the empirical frequency of that sequence in the corpus. The probability of a corpus under a bigram model is the product of the probability of each character in the corpus given the character that precedes it.

Let our corpus be a string of symbols from the set of phonemes of the language + '#', the word boundary symbol. Let the corpus begin and end with '#' and contain no '##' sequences. Then a word in the corpus is a substring that begins and ends with '#' and has no internal occurrences of '#'. Let $|c|$ be the length of the corpus and $c[i]$ be the ith letter in the corpus. Let $Count(a)$ be the number of occurrences of a in the corpus. Then the probability of the corpus of the unigram model is[2]:

$$p(c) = \prod_{i=2}^{|c|} p(c[i]) \qquad p(a) = \frac{Count(a)}{|c|} \tag{7}$$

Consequently, the description length of the corpus is:

$$plog(c) = \sum_{i=2}^{|c|} plog(c[i]) \tag{8}$$

Now let $Count(ab)$ be the number of occurrences of a immediately followed by b in the

[2] The unigram model skips over the initial word boundary. This is because the corpus is best thought of as a circle. Words are strung together with boundary symbols between them until you eventually loop around and end up where you started. Only to store and process this, we pick an arbitrary word boundary symbol and cut the circle there, splitting that word boundary symbol so that it shows up at both ends. So the first and last word boundary are *the same word boundary.* It would be an error to count it twice

corpus. Then the probability of the corpus under a bigram model is:

$$p(c) = \prod_{i=2}^{|c|} p(c[i]|c[i-1]) \qquad p(b|a) = \frac{Count(ab)}{Count(a)} \tag{9}$$

Consequently, the description length of the corpus under a bigram model is:

$$plog(c) = \sum_{i=2}^{|c|} plog(c[i]|c[i-1]) = \sum_{i=2}^{|c|} plog(c[i-1]c[i]) - plog(c[i-1]) \tag{10}$$

A crucial information theoretic concept is *Mutual Information*. The Mutual Information between two characters a and b is defined in (11).

$$MI(a, b) = plog(a) + plog(b) - plog(ab) \tag{11}$$

Intuitively, the Mutual Information is the number of bits saved using a bigram encoding for ab instead of a unigram encoding. Summing the Mutual Information between each pair of characters in the corpus gives the difference between the unigram and bigram code lengths.

The Mutual Information can be thought of as a force two segments exert on each other. If $MI(a, b)$ is positive, then a and b attract. If it is negative, a and b repel. The Boltzmann models used here incorporate MI from distant vowels into the model to try to shorten the description length of the corpus.

4.2 Autosegmental models and Boltzmann distributions

In vowel harmony languages, a great deal of information is shared between vowels. When the first vowel in the word is chosen, it nearly cuts in half the possibilities for the subsequent vowels. This sort of information can't be captured in a bigram language model.

Rather than going to trigrams or 4-grams, we posit an abstract representation that separates the vowels from the rest of the word on their own "plane", where they interact locally. This autosegmental representation (Goldsmith, 1990) allows us to capture vowel harmony as a local process between adjacent vowels. An autosegmental representation with the vowels on their own tier separate from the bigram tier is preferable to a trigram or

4-gram or arbitrary n-gram model for two reasons. First, the trigram or 4-gram model will still miss V-to-V interactions if they are separated by two or three (or more) consonants. Second, if a language has P phonemes, the trigram model will add P^3 parameters to the bigram model. The 4-gram model adds P^4 parameters to the trigram model. This greatly complicates our language model and increases the length of the grammar. Worse, it makes it difficult to accurately estimate the model's parameters, even from generously large corpora. In contrast, if a language has V vowels, an autosegmental vowel tier adds V^2 parameters to the model, and so the overall grammar will be simpler than a trigram model[3].

Incorporating the V-to-V information in the vowel tier into a language model is no simple task. Removing the vowels from the timing tier occludes local C-to-V and V-to-C, effects, which Goldsmith and Riggle show to be *more* significant than nonlocal V-to-V effects in Finnish. To overcome this, they use a Boltzmann distribution to incorporate V-to-V information.

The Boltzmann distribution (also known as the Gibbs distribution, and some other names besides) arose in statistical physics and thermodynamics, but it's not completely new to linguistics. Geman and Johnson (2001) give an introduction to its linguistic applications. Galves and Galves (1995) used a Boltzmann distribution to model performance and learning effects that they use to account for a historical shift in clitic placement in European Portugese.

The Boltzmann distribution takes a markedness score and defines a probability distribution from it, with higher markedness corresponding to lower probability. If w is a word from our corpus and s is a scoring function that assigns a markedness rating to every possible word, then the probability of w defined by the Boltzmann distribution is (12).

$$p(w) = \frac{2^{-s(w)}}{Z} = \frac{2^{-s(w)}}{\sum\limits_{x \in S} 2^{-s(x)}} \tag{12}$$

Z here is a normalizing constant that is gotten by summing all of the pseudo-probability assigned by $2^{-s(x)}$ to every string of phonemes x in the sample space S.

The plog of a word under the bigram model is an appropriate scoring function for s. In

[3] Incorporating an autosegmental tier complicates our representation and the method used to calculate a word's probability, and this should be factored in to the length of the grammar when doing an MDL analysis. This additional complexity is still dwarfed by the complexity of storing all the trigrams in a trigram model and their codewords.

fact, using the bigram plog of a word as the scoring function for a Boltzmann distribution yields exactly the bigram model. The power of the Boltzmann distribution is its ability to incorporate other sources of markedness as well. By separating the vowels from the corpus into a vowel tier and counting the vowel bigrams, we can compute the V-to-V MI in the vowel tier and incorporate than into our scoring function s.

Recall that if a and b have a positive MI, they attract, meaning ab sequences are less marked than it would be taking the individual markedness of a and b alone. The higher the MI, the more they attract. So to turn this into a markedness score, we must *subtract* the V-to-V MI from our bigram model markedness score. This gives us the scoring function s shown in (13).

$$s(w) = plog_{Bigram}(w) - MI_V(w) \tag{13}$$

Goldsmith and Riggle show that the adjacent vowels in a word differ in their distribution from successive vowels with intervening consonants. Rather than counting the MI between every pair of vowels in the corpus, they count MI between only non-local pairs of successive vowels[4]. Local VV effects are already counted in the bigram model, and since non-local VC$^+$V interactions are different from local VV interactions, counting only non-local vowel bigrams in the vowel tier produces a tighter fit with the data. For these reasons and to compare my results to theirs, I also counted only non-local vowel bigrams in the vowel tier.

When working with a Boltzmann distribution it can be difficult, and sometimes even impossible to compute the normalization constant Z. But the scoring function s we're working with can be represented as a weighted finite state automaton (WFSA), from which we can compute Z using a dynamic programming technique in Eisner (2002).

Start by representing the bigram model as a WFSA. Each state corresponds to a character

[4] The definition of Mutual Information must be changed slightly from the one given in (11) when counting only non-local vowels. For vowels a and b in a $V_1C^+V_2$ configuration, $MI(a,b) = plog(a_l) + plog(b_r) - plog(ab)$, where $p(a_l)$ is the number of times a occurs as V_1 in the $V_1C^+V_2$ environment over the number of times that environment occurs. Likewise, $p(b_r)$ is the number of times b occurs as V_2 in the environment $V_1C^+V_2$, divided by the number of times that environment occurs.

This definition of MI is more general than the one given in (11). Normally when counting bigrams, every occurrence of a character a is the first letter of on bigram and the second letter of another. In this case, $plog(a_l) = plog(a_r) = plog(a)$, and the definition in (11) is correct. This is not usually the case when considering only non-local VV bigrams.

The Python source code used to count $V_1C^+V_2$ bigrams and compute V-to-V mutual information is given in appendix B.

in the alphabet. The arc between states emits the character of the state it's moving in to, and has a weight equal to the conditional plog of the character who's state it's moving into, given the previous character (whose state it's coming from). Then, construct a WFSA representation of the non-local vowel model[5]. Use WFSA intersection (Mohri, 2004) to get a WFSA representing both the bigram model and the vowel model acting on a word simultaneously. The weights on the arcs of this WFSA are additive, and the WFSA now represents the scoring function s used for our model. Exponentiate the weights to get a model whose arc weights are multiplicative, so that the model assigns pseudo-probabilities to words. Now you can step through the new machine. At each step, take the states you were able to reach at the last step and find out which states can be reached from them. Multiply the total weight in the state from the previous step by the weight on the arcs to the next states, and for each state reachable, keep a running total on the weights going in to that state. Run this repeatedly until the weight on the final word boundary state stops increasing (which is not guaranteed to happen, in which case the model is not normalizable). The total weight at the final state is the value of Z.

4.3 Vowel Mutual Information results

I constructed unigram, bigram, and Boltzmann models for the same Finnish, Turkish, Italian, and English corpora used for the HMM experiments in section 3. For each language, I constructed two Boltzmann models: one without word boundaries in the vowel tier, following Goldsmith and Riggle (2007), and a second with word boundaries in the vowel tier.

We would expect that vowel harmony languages would have a great deal more information shared between distant vowels than non-vowel harmony languages. Therefore, we should see a greater reduction in the number of bits used to encode the corpus when moving from the bigram to the Boltzmann model when moving from vowel harmony languages compared to non-vowel harmony languages. Further, if we figure in the $plog(g)$ term in equation (6), we would expect the MDL principle to accept the addition of a vowel tier to the model in vowel harmony languages and reject its addition in non-vowel harmony languages. In fact, this is exactly what we find.

[5] The WFSA representations I used for the vowel tier are provided in appendix C.

4.3.1 Finnish

I start by replicating the findings of Goldsmith and Riggle using the same Finnish corpus. Table 2 shows the numbers they published for their Finnish models.

Unigram Model:	2,088,530 bits
Bigram Model:	1,780,621 bits
Boltzmann Model:	1,760,501 bits
Z:	1.0123
Boltzmann/Bigram:	0.988700571317535

Table 2: Bit cost for each model from Goldsmith and Riggle (2007)

My own numbers, shown in Table 3 were very close but not quite identical. They are close enough that any error in the computation of bit length is so small that it should not effect the overall results much.

Unigram:	2,088,534 bits
Bigram:	1,780,262 bits
Boltzmann:	1,760,517 bits
Z:	1.00254060086

Table 3: Computed bit cost for the same corpus used in Goldsmith and Riggle (2007).

Before looking at the V-to-V MI and its effect on the representation length of the Finnish corpus, we should remind ourselves what we should expect. Vowel harmony should very strongly penalize disharmonic pairs of vowels, producing very low (high negative) MI between disharmonic vowels. Harmonic pairs of vowel should be preferred, or at least not be dispreferred nearly as strongly as disharmonic pairs. The neutral vowels [i, e] should be ok with any of the other vowels. In fact, we might expect the neutral vowels and the other vowels to resemble V-to-V interactions in non-vowel harmony languages. Looking at Table 4, we see that the neutral vowels have very low magnitude interactions with all the other vowels, which is the same trend we'll see when we turn to Italian V-to-V interactions.

Table 4 shows exactly what we expect. The upper left and lower right side of the table are the harmonic vowel interactions. The lower left and upper right side are the disharmonic vowel interactions. Notice that the upper right and lower left corners have all negative MI

	y	ä	ö	e	i	o	u	a
y	2.00	1.74	1.75	0.26	0.17	-1.73	-2.94	-2.42
ä	1.16	1.85	2.68	0.03	0.33	-2.25	-2.29	-2.39
ö	1.57	1.63	2.03	-0.20	-0.43	-0.45	-0.43	-0.95
e	0.31	0.64	-0.87	-0.40	0.24	-0.02	0.20	-0.31
i	0.21	0.05	0.30	0.29	-0.47	0.01	-0.04	0.04
o	-1.58	-2.63	-4.12	0.18	0.18	0.25	-0.30	0.20
u	-1.45	-2.42	-3.42	0.12	-0.13	-0.01	0.58	0.23
a	-2.03	-2.73	-4.21	-0.46	0.16	0.33	0.17	0.42

Table 4: Finnish V-to-V MI in VC$^+$V environment

values. In fact, most of those values are much lower than any other values on the table. This captures the fact that disharmonic vowels are very marked, so any form containing a sequence of disharmonic vowels will pay a steep penalty.

Notice also that the upper left corner shows very high MI between the front vowels. Finnish front vowels very strongly select for other front vowels and tolerate co-occurrence with the neutral vowels [i,e]. The back vowels, on the other hand, do not select for each other nearly so strongly, and in the case of [o] and [u], actually repel slightly. Their repulsion is not nearly as strong as most of the disharmonic pairings, and even an *ou* sequence is less repulsive than an disharmonic pairing.

Table 4 shows that vowel selection is very uneven in Finnish. Within harmony classes we see a range of preferences for various vowel sequences, and certain disharmonic pairings, such as an [ö] followed by a back vowel, are much more tolerated than others.

The number of bits used to encode the Finnish corpus by each model are shown in Table 5 along with the bits used to encode the grammar and the total description length of the corpus and the grammar for each model considered. The Vowel+# Boltzman model is the same as Boltzmann model with the autosegmental vowel tier, but with the word boundaries included on the vowel tier. Doing this will always decrease the description length of the corpus because the word boundaries do not block any potentially information rich V-to-V interactions that the model without the word boundaries captured, and it will substantially decrease the encoding length of the corpus if some vowels prefer or avoid being the first or last vowel in the word. But adding word boundaries to the vowel tier failed to really substantially lower the description length. Adding the vowel tier added 64 parameters to the model and reduced the description length of the corpus by just over 1%. Adding word boundaries to

Model Name	Corpus DL	Grammar DL	Total DL
Unigram:	9,850,123 bits		
Bigram:	8,427,638	61,055	8,488,692
Vowel Boltzman:	8,342,998	99,195	8,442,192
Vowel+# Boltzmann:	8,329,600	101,438	8,431,038

Model Ratios:	Corpus DL ratio	Total DL ratio
VMI/Bigram:	0.989956880322	0.994522138948
V#MI/Bigram:	0.988367125479	0.993208058281
V#MI/VMI:	0.998394117083	0.998678681334

Table 5: Bit cost for each Finnish model

the vowel tier requires 17 additional parameters a, but only reduces the description length of the corpus an additional 0.1%. However, those 17 extra parameters add very little in the way of complexity compared to the original 64 parameters, mainly because adding the 64 vowel parameters to the model also introduce what we could loosely call a new way of counting voilations, namely by skipping over consonants. The extra 17 parameters added when adding word boundaries use the same method to count voilations as vowels, and so add less complexity to the system when they are added. Table 6 shows V-to-V interactions with word boundaries counted as a vowel.

	y	ä	ö	e	i	o	u	a	#
y	1.87	1.77	1.96	0.40	0.32	-1.74	-3.02	-2.39	-0.66
ä	0.91	1.77	2.77	0.05	0.37	-2.37	-2.49	-2.47	0.41
ö	1.32	1.54	2.12	-0.18	-0.40	-0.57	-0.63	-1.04	0.45
e	-0.13	0.35	-0.98	-0.58	0.07	-0.35	-0.19	-0.60	1.27
i	0.05	0.06	0.49	0.40	-0.35	-0.01	-0.14	0.04	-0.30
o	-1.73	-2.61	-3.93	0.30	0.31	0.23	-0.40	0.21	-0.40
u	-1.59	-2.40	-3.23	0.24	0.00	-0.03	0.49	0.24	-0.43
a	-2.24	-2.79	-4.09	-0.41	0.23	0.24	0.00	0.36	0.21
#	0.74	0.26	-0.71	-0.23	-0.31	0.39	0.61	0.26	-4.86

Table 6: Finnish V-to-V MI in VC$^+$V environment, with word boundaries

The non-local V-to-# interactions are all low magnitude, much like the interactions between the neutral vowels and the other vowels. Whether word boundaries should be included in a vowel tier will depend a great deal on whether other aspects of a more complete phonological model capture these effects (which would render the V-to-# effects seen in Table

6 epiphenomenal).

The V-to-V MI in our Boltzmann model adequately captures many aspects of Finnish vowel harmony. It allows us to group the Finnish vowels into harmonic classes that have relatively high MI for V-to-V interactions within the class and very low MI for V-to-V interactions across classes. It also captures some of the neutrality of [i,e]. It fails to capture the fact that vowel harmony occurs through the neutral vowels though. In this model, a neutral vowel will block the transmission of potentially high-information interactions between harmonic vowels.

One solution is to remove the neutral vowels from our vowel tier. To do this we'd need some criteria for identifying neutral vowels. The HMM algorithm discussed in section 3 gives us a criteria. If the penalty you'd pay for emitting a vowel from its dispreferred state is less than the penalty for changing states, then that vowel is a neutral vowel. Table 7 shows that removing neutral vowels from the vowel tier not only simplifies the model, but lowers the corpus description length by allowing distant V to V interactions pass through the neutral vowels.

Model Name	Corpus DL	Model DL	Total DL
Vowel MI	8,317,291	89,965	8,407,256

Ratio	Corpus DL ratio	Total DL ratio
VMI/Bigram	0.986906511059	0.990406449324

Table 7: Bit cost Finnish vowel model without neutral vowels

Another solution would be to put the vowel categories into the vowel tier instead of the vowel themselves. The categories could be tagged by having an HMM trained as in section 3 find the best parse for the word, using the states in that best path for categories. This would have the added benefit of introducing fewer parameters into the model, thus simplifying the grammar. The downside is that it would fail to capture some of the idiosyncrasies within the vowel classes, and it's also not clear whether the HMM used would have to remain as part of the description length of the grammar, or whether the tags it applies can be stored in the grammar and the HMM could subsequently be discarded.[6]

[6] Clearly some method would have to be retained for tagging novel words, but once a large enough seed of tagged words is created, a cheaper way of tagging novel words could be found.

4.3.2 Turkish

The Turkish vowel facts reviewed in section 2.2 would lead us to expect that any V-V sequence that disagrees in front/backness should have very low MI. High MI should occur amongst vowels that agree in both front/backness and roundness, and roundedness disharmony should be costlier when the second vowel in the sequence is a high vowel than it is when the second vowel is low. Table 8 shows the observed V-V MI.

	i	e	ü	ö	ı	a	u	o
i	0.69	0.71	-2.34	-0.73	-6.81	-0.79	-3.32	0.22
e	0.89	0.79	-2.06	0.26	-4.42	-1.86	-2.52	-0.32
ü	-2.51	0.99	3.77	0.54	-5.31	-1.88	-2.11	-0.82
ö	-4.20	1.33	3.65	1.57	-6.66	-3.74	-3.64	-1.12
ı	-4.26	-3.17	-4.83	-0.99	1.89	0.84	-4.94	-1.29
a	-0.52	-1.61	-2.30	0.34	1.16	0.59	-1.54	-0.04
u	-2.84	-2.04	-2.02	-2.19	-5.98	0.77	2.90	-0.72
o	-0.91	-0.98	-0.88	-0.08	-5.17	0.43	2.24	1.44

Table 8: Turkish V-to-V MI in VC$^+$V environment

First, the gross trends. The lower left and upper right corners are disharmonic front-back pairings, and are almost have negative MI (with the exception of the [aö] sequence). The 2-by-2 squares along the diagonal of the table are all harmonic with respect to both backness and roundness, and all the values in these squares are positive and generally much higher than the other values, with the exception of the sequence [uo].

More specific claims about Turkish vowel harmony seem to borne out in the data in Table 8. High vowels have a stronger tendency to harmonize for roundness than low vowels. Front round vowels followed by [i] have low MI, but have much higher MI when follewd by [e]. All cases of high vowels followed by roundness-disharmonic high vowels have negative MI, low-low, high-low, and low-high frequently have positive MI. Table 8 also confirms Kirchner's observation that [ı] does not occur in either front/back or round/unround disharmonic configurations; the MI values between [ı] and any disharmonic segments are the lowest values on the table.

But while the marked segments [üö] tend to avoid front/back disharmony, they don't seem to do so with any more regularity than any of the other segments. The actual facts of Turkish vowel interaction are more complicated than a rule governed picture of vowel harmony makes them out to be, with lots of partial dependencies and preferences, and the Boltzmann model allows us to capture those facts.

Model Name	Corpus DL	Model DL	Total DL
Unigram	5,109,878		
Bigram	4,250,794	74,428	4,325,222
Vowel MI:	4,048,875	116,582	4,165,457
Vowel w/# MI	4,029,733	119,070	4,148,802

Ratio	Corpus DL ratio	Total DL ratio
VMI/Bigram	0.952498433713	0.963062012551
V#MI/Bigram	0.947995330581	0.959211506645
V#MI/VMI	0.995272324895	0.996001808964

Table 9: Bit cost for each Turkish model

Table 9 shows the bit cost of encoding the Turkish corpus under the models I've tested it on. Moving from the bigram model to the Boltzmann model with long distance V-to-V information allows us to shorten the description of the Turkish lexicon by nearly 5%. This is a much larger savings than we got by including V-to-V information in the Finnish model, probably due to the fact that Turkish has no neutral vowels and possibly because Turkish words may contain more vowels than Finnish words. Very few of the Turkish V-to-V interactions were near chance (0 MI), while quite a few of the V-V interactions involving neutral vowels were.

Table 9 also shows nearly a 0.5% model improvement when we add word boundaries into the vowel tier. While the extra boundary parameters don't capture as much as the V-to-V parameters, they still capture more information than the boundary parameters captured in Finnish. Table 10 shows the the vowel tier MI with word boundaries added to the vowel tier.

Table 10 reveals some interesting generalizations about vowels and word edges. The low round vowels [öo] prefer to be the first vowel in the word and tend to reject following another vowel. Conversely, the unround vowels [ieıa] show a weaker tendency to occur at the end of a word. To my knowledge, these sorts of facts are not ordinarily noticed by phonologists.

	i	e	ü	ö	ı	a	u	o	#
i	0.75	0.62	-2.77	-2.73	-6.67	-0.96	-3.55	-0.43	0.60
e	1.01	0.76	-2.43	-1.67	-4.22	-1.96	-2.69	-0.92	0.29
ü	-2.30	1.06	3.50	-1.30	-5.01	-1.89	-2.18	-1.32	-0.44
ö	-3.92	1.47	3.44	-0.20	-6.29	-3.68	-3.64	-1.55	-1.43
ı	-4.24	-3.29	-5.30	-3.03	1.99	0.63	-5.21	-1.99	0.76
a	-0.36	-1.60	-2.63	-1.55	1.40	0.53	-1.67	-0.60	0.04
u	-2.70	-2.05	-2.37	-4.11	-5.76	0.68	2.75	-1.30	0.18
o	-0.71	-0.94	-1.18	-1.95	-4.90	0.39	2.14	0.91	-0.20
#	-0.68	0.09	0.97	2.06	-1.50	0.38	0.55	1.29	-3.78

Table 10: Turkish V-to-V MI in VC$^+$V environment, with word boundaries

4.3.3 Italian

Italian does not have vowel harmony, so we expect that incorporating V-to-V MI into a model of Italian would not substantially reduce the description length of our corpus, especially in comparison to the savings we've seen in Turkish and Finnish. Table 11 shows Italian V-to-V MI is very close to zero, so distant vowel to vowel interactions are at near chance.

	e	i	a	o	u
e	0.08	-0.07	0.10	-0.16	0.05
i	-0.09	-0.12	0.20	-0.03	0.25
a	-0.05	0.08	-0.31	0.29	-0.12
o	0.19	0.01	-0.00	-0.25	-0.19
u	-0.38	0.22	0.21	-0.29	-0.04

Table 11: Italian V-to-V MI in VC$^+$V environment

As we expect, incorporating this information into our language model does not yield a substantial savings (less than 0.1%). Incorporating word boundaries helps, but the overall savings we get by adding a vowel tier with word boundaries to the bigram model is still less than a 0.2% reduction in the size of the corpus.

Notice that the total description length of the vowel model is larger than the bigram model. The MDL principle rejects this model; the extra complexity of the model is not justified by a correspondingly large improvement of the fit of the model to the data.

Name	Corpus DL	Model DL	Total DL
Unigram	2,799,560		
Bigram	2,310,130	51,416	2,361,545
Vowel MI	2,308,321	74,045	2,382,366
Vowel w/# MI	2,305,799	76,094	2,381,893

Ratio	Corpus DL ratio	Total DL ratio
VMI/Bigram	0.999217066766	1.00881647537
V#MI/Bigram	0.998125220863	1.00861604908
V#MI/VMI	0.998907298584	0.99980132532

Table 12: Bit cost for each Italian model

4.3.4 English

The lack of savings could in part have to do with Italian's relatively small vowel inventory compared to Finnish or Turkish. English, however, has a much larger vowel inventory than any of these languages. Adding its vowel information to the bigram model still only yields less than a 0.3% reduction of the corpus description length, and increases the total description length when the complexity of the model is included.

Name	Corpus DL	Model DL	Total DL
Unigram	4,650,896		
Bigram	3,823,409	107,736	3,931,145
Vowel MI	3,813,998	206,099	4,020,097
Vowel w/# MI	3,752,375	209,307	3,961,682

Ratio	Corpus DL ratio	Total DL ratio
VMI/Bigram	0.997538466872	1.02262769802
V#MI/Bigram	0.981421212678	1.00776797985
V#MI/VMI	0.983842974753	0.98546908303

Table 13: Bit cost for each English model

Interestingly, adding word boundaries to the vowel tier yields an improvement greater than adding the vowel tier to the bigram model. Table 14 shows the V-to-V MI with word boundaries. Like Italian, and quite unlike Finnish or Turkish, most of the vowels show only low-magnitude interactions with each other.

	AA	AE	AH	AO	AW	AY	EH	ER	EY	IH	IY	OW	OY	UH	UW	#
AA	-0.0	-2.1	0.5	-1.0	-1.4	-1.5	-0.3	0.0	-0.6	-0.1	0.5	0.9	-0.8	-0.5	-0.4	-0.2
AE	-1.8	-1.4	0.8	-1.0	-0.9	-0.9	-1.1	0.6	-1.4	0.3	0.0	-0.3	-0.5	-0.6	-0.4	-0.3
AH	-0.7	-0.8	-0.7	-0.4	-0.5	0.1	-0.3	-0.4	0.4	-0.4	-0.2	-1.1	-0.2	-1.1	-0.4	0.9
AO	-1.5	-1.5	0.3	-0.9	-0.3	-1.2	-0.6	0.3	-0.7	-0.2	0.7	-0.7	-1.5	-0.8	-1.6	0.3
AW	-2.0	-1.9	-0.0	-1.3	0.1	-0.3	-1.4	0.9	-0.1	-0.2	0.0	-0.9	-0.5	-0.4	-2.3	0.6
AY	-1.2	-1.2	0.1	-0.9	-0.4	-0.7	-1.4	0.9	-0.8	-0.1	-1.2	-0.4	-1.1	-2.2	-2.5	0.7
EH	-1.3	-2.1	0.7	-1.1	-1.2	-1.3	-1.2	0.6	-1.2	0.0	0.4	0.1	-0.0	-0.7	-1.0	-0.1
ER	-1.3	-1.0	-0.1	-0.7	-0.3	-0.0	-0.9	-0.4	-0.3	-0.5	-0.3	-1.1	-1.5	-0.0	-0.9	0.9
EY	-1.6	-2.9	0.7	-1.7	-0.9	-2.3	-2.2	0.7	-1.2	-0.1	0.0	-0.6	-2.2	-1.1	-2.1	0.3
IH	-1.2	-1.4	0.2	-0.8	-1.1	-1.5	-0.8	0.2	-1.1	-0.5	-0.1	-0.7	-1.0	-0.9	-0.9	0.7
IY	-0.8	-1.2	0.0	-0.6	-1.2	-0.9	-0.8	0.0	-0.6	-0.8	0.3	0.7	-1.7	-0.6	-1.1	0.5
OW	-0.1	-0.9	0.1	-1.1	-0.8	-0.9	-0.3	0.6	-0.1	-0.5	0.6	0.2	-1.0	-0.6	-1.4	0.0
OY	-1.8	-2.4	0.1	-1.8	-1.6	-2.8	-1.7	0.6	-1.7	-0.4	-0.8	-1.4	–	-2.2	-2.6	1.0
UH	-0.7	-1.5	0.1	-0.9	-0.3	-0.7	-0.7	-0.0	-1.1	-0.0	0.0	0.1	-1.6	-1.2	-1.7	0.5
UW	-0.5	-1.3	0.4	-1.3	-0.9	-1.3	-0.8	0.4	-0.2	-0.0	0.5	0.1	-0.6	-1.3	-0.9	-0.0
#	1.0	1.2	-0.5	1.0	1.0	0.9	0.9	-0.7	0.6	0.4	-0.2	0.4	0.9	1.0	1.0	-12.

Table 14: English V-to-V MI in VC$^+$V environment, with word boundaries

The improvement in the description length when adding word boundaries is very likely epiphenomenal. It's probably caused by an interaction between stress placement (which interacts with word boundaries) and vowel reduction.

4.4 Conclusion

The information theoretic approach using an autosegmental vowel tier allows us to quantify the magnitude of long-distance vowel-to-vowel interactions within a language. I have shown that for the four languages I've looked at, the magnitude of those interactions is much greater when that language is a vowel harmony language. In addition, the MDL principle provides a precise criteria that justifies modeling distant V to V interactions in vowel harmony languages but not in non-vowel harmony languages.

This technique has some advantages and some disadvantages compared to the HMM technique. The advantage is its granularity. It's able to model a wider range of interacions than an HMM because those interactions occur between the segments themselves, and not the abstract categories to which the segments belong. It also has fewer parameters to estimate than the HMM, and parameter estimation is orders of magnitude faster and not subject to variation based on an initial model.

The drawbacks of the IT model is that it doesn't categorize the segments into natural classes. That must be done by something else, either outside observation or some sort of clustering algorithm. A related issue is the way it handles segments that are neutral to vowel harmony. The HMM algorithm allows vowel harmony information to pass through neutral vowels. The IT model does not.

5 The Big Picture

The previous discussion of HMMs and Boltzman models focused on their usefulness as tools for dectecting vowel harmony patterns already noted by linguists. But there's more at work here than simply quantifying known phenomena. The introduction to section 4 puts forward a Bayesian conception of natural language grammar. Under this new conception, a phonology is a set of forces of phonemes acting on each other that partially determines how a word is pronounced.

But this new grammatical framework introduces a number of problems that need to be worked out.

5.1 Other phonotactic effects: syllabification and stress

One promising area of future research is syllable structure. There's a great deal of regularity in the syllable structure that's not exploited in the simple bigram model. Some segments are much more marked as coda segments than others, and likewise some onsets are more marked than others. These factors can vary quite a bit from language to language, which makes it particularly challenging to infer them from unlabeled data.

Larson (1993) gives a connectionist method for inferring syllable structure from input words labeled with syllable peaks. Section 3 shows that HMMs can pick out vowels from consonants. The vowels in the most probable HMM parse of a word can serve as the syllable peak labels for the connectionist learning algorithm.

But learning syllable structure is not the same as incorporating that syllable structure into the grammar. One way to incorporate syllable structure would be to count all the distinct onsets that occur in the language. The probability of each onset will be the number of times it occurs in the corpus divided by the total number of onsets that occur in the corpus. The same can be done for codas, and if this process can be given a finite state representation[7], it can be integrated into the Boltzmann model the same way distant vowel information was and we can measure how much syllable phonotactics reduces the description length of the grammar.

Stress patterns are also readily learnable from surface phonology, though they present a challenge for automatic language learning from text because many corpora do not have stress marked. Larson's connectionist model is able to learn stress patterns. Probabilistic finite state methods can also be used to infer stress systems.

Once the stress system is inferred, it's interaction with vowel quality and weight is just another form of mutual information. Mutual information between stress and vowel quality can model reduction of unstressed vowels. MI between stress and codas can model weight effects on stress assignment.

5.2 Phonological alternations

Generative phonology has taken phonological alternation, more than anything else, as the thing to be explained by phonology. I have not addressed phonological alternation, but we can imagine at least two approaches within an information theoretic framework.

One method is to relate surface forms through morphology. This would involve constraints saying that morphologically related forms should also be phonologically related. Pronunciation differences between morphologically related forms would be penalized by a drop in probability. Phonological alternations would occur when the penalty for dissimilarity among related words is less than the penalty for a marked structure. This sort of approach is essentially the same as Output-Output constraints in Optimality Theory.

The challenge for this theory is unsupervised learning of morphology and learning the penalties for dissimilarity. Goldsmith (2001) offers one approach to the first problem. In the absence of a good unsupervised morphology learning algorithm corpora with word lemma

[7] An issue that might make such a finite state representations difficult to create is smoothing. No possible string should receive 0 probability, so a portion of the prabability mass must be assigned to unobserved events. Assigning 0 probability to a word doesn't mean the word is merely impossible; it means the word is *unlearnable.*

tags are readily available for a number of languages. Once morphologically related forms are identified we might try setting the probability of dissimilarity by counting the number of morphologically related forms that show the same dissimilarity and dividing by the total number of relevant morphological forms.

Another approach is to posit underlying representations. Each segment in the underlying representation would exert a force on the surface correspondent to retain its underlying character, but that force could be overcome by the environmental pressure to have a different character that forms a less marked structure. The connection between the underlying representation and its surface form is just another form of mutual information. Vowel harmony offers one example. The large difference in mutual information between a disharmonic pair of vowels and a harmonic pair will pressure a suffix vowel to surface as a harmonic vowel, overcoming the MI shared between the underlying representation and it's favored surface correspondant.

The challenge with this approach is to learn the underlying representations from the surface forms. Billerey-Mosier (2003) presents a method for inferring an underlying representation from the surface form, and something like this method might work in the information theoretic framework.

This picture of surface segment-to-segment information competing with underlying representation to surface form information is very similar to the OT picture of markedness constraints competing with faithfulness constraints. In fact, it bears an even stronger similarity to Harmonic Grammar (Goldsmith, 1993). Harmonic Grammar assigns a real valued weight to constraints, rather than strictly ranking them. A constraint violation in a harmonic grammar incurs a penalty equal to the weight of the constraint violated, and the surface form is the candidate that accrues the least penalty. Any sort of mutual information a language model incorporates can be thought of as a harmonic grammar constraint, with its weight equal to -1 times the MI value.

5.3 Computing grammar length and adding parameters

The question of how to assign a probability to a grammar looms largo over this entire enterprise. Most MDL algorithms, including Goldsmith (2001) and de Marken (1996), compare grammars that vary with the parameters they use but are fixed with the respect to the algorithms that make use of those parameters. When this is the case, the MDL analysis can assign a length of the grammar proportional to the code lengths of the parameters.

In an information theoretic model, the algorithm for using some parameters is different than the algorithm for using others, and the complexity of those different algorithms must also be considered as a factor. Goldsmith (2007b) argues for one way to measure the complexity of an algorithm: the length of the source code for the shortest implementation of that algorithm on a universal Turing machine. This opens an important research agenda for the information theoretic framework: devise a universal Turing machine appropriate for natural language grammar.

The problem is important for unsupervised learning because we want the grammar to steadily grow as it gets more data. Extra data will justify the additional complexity in the grammar if and only if that extra complexity leads to a decrease in the number of bits needed to encode the data outweighing the increase in the number of bits needed to encode the grammar.

6 Conclusion

This paper has shown two things. First, it's shown that the methods in Goldsmith and Xanthos (2006) and Goldsmith and Riggle (2007) both capture vowel harmony facts in languages with vowel harmony without spuriously identifying vowel harmony in languages without it. Second, I've shown that, from an information theoretic perspective, the distant vowel-to-vowel interactions in vowel harmony languages are statistically important effects that should be captured by the phonology, while simultaneously showing that similar interactions in non-vowel harmony languages are less significant, with reason to doubt that they should also be captured by the phonology.

A Grammar Complexity

The models used are deterministic finite state machines. The total cost of a grammar is simply the sum of lengths of the encoding of the component finite state machines, plus a small overhead. The total cost function is summarized below.

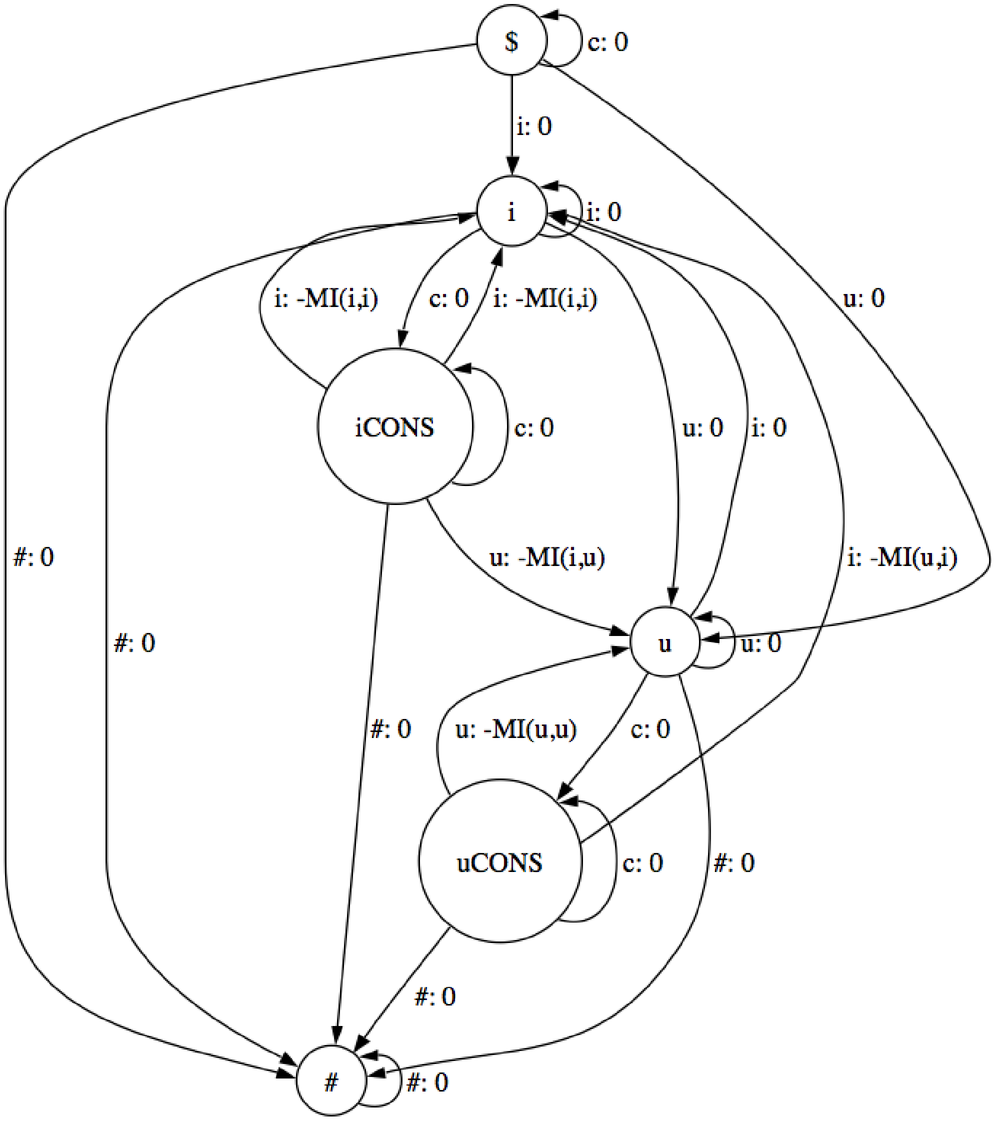
- All scoring functions are finite state string to weight transducers.
- First state is assumed as the only start state (start weight 0). All states final.
  - $s = \#$ of states,
  - $a = \#$ of arcs,
  - $l = \#$ of letters in the alphabet
- $L = a(2log_2(s) + log_2(l) + 64) + log_2(s)$
- Small added overhead to mark where one transducer ends and the next begins: $8\lceil log_{127}(L)\rceil$

B Counting Non-Local Vowel Bigrams

```
def nonlocal_v_mi_by_word( corpus, vowels ):
    #corpus is a list of letter sequences, each sequence beginning and
    #ending with '#' and containing no internal '#'s
    left_unigram_plog = {}
    right_unigram_plog = {}
    bigram_plog = {}

    for letter in vowels:
        left_unigram_plog[letter] = 0
        right_unigram_plog[letter] = 0
        for second in vowels:
            bigram_plog[(letter, second)] = 0

    for word in corpus:
        last = None
        last_vowel = None
        for letter in word:
            if last == None:
                last = letter
                if letter in vowels:
                    last_vowel = letter
                continue
            if letter in vowels:
                if last_vowel == None or last == last_vowel:
                    last = letter
                    last_vowel = letter
                    continue
                #current letter is a vowel and the last vowel was not adjacent
                bigram_plog[(last_vowel, letter)] +=1
                last_vowel = letter
            last = letter

    for bigram in bigram_plog.keys():
        left_unigram_plog[bigram[0]] += bigram_plog[bigram]
        right_unigram_plog[bigram[1]] += bigram_plog[bigram]

    num_l_unigrams = float(sum(left_unigram_plog.values()))
    num_r_unigrams = float(sum(right_unigram_plog.values()))
    for letter in left_unigram_plog.keys():
        left_unigram_plog[letter] = \
            -log(left_unigram_plog[letter]/num_l_unigrams, 2)
        right_unigram_plog[letter] = \
            -log(right_unigram_plog[letter]/num_r_unigrams, 2)

    num_bigrams = sum(bigram_plog.values())
    for bigram in bigram_plog.keys():
        if bigram_plog[bigram] == 0:
            bigram_plog.pop(bigram)
        else:
            bigram_plog[bigram] = -log(bigram_plog[bigram]/float(num_bigrams), 2)

    mi = {}
    for bigram in bigram_plog.keys():
        mi[bigram] = left_unigram_plog[bigram[0]] + right_unigram_plog[bigram[1]] \
            - bigram_plog[bigram]
    return mi
```

C Weighted Finite State Automata Representation of Vowel MI

Below is a graph showing the weighted finite state machine representing the vowel tier of the Boltzmann model for a toy language with an alphabet of [c,u,i] and vowels [u,i].



References

Billerey-Mosier, Roger. 2003. Exemplar-based phonotactic learning. In *SWOT*.

Eisner, Jason. 2002. Parameter estimation for probabilistic finite-state transducers. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Ellison, Mark T. 1994. The iterative learning of phonological constraints. *Computational Linguistics* 20.

Galves, A, and C Galves. 1995. A case study of prosody driven language change. Unpublished Draft.

Geman, Stuart, and Mark Johnson. 2001. Probability and statistics in computational linguistics, a brief review. URL `http://www.cog.brown.edu/mj/papers/Review.pdf`, manuscript.

Goldsmith, John. 1990. *Autosegmental and metrical phonology*. Oxford: Blackwell.

Goldsmith, John. 1993. Harmonic phonology. In *The last phonological rule*, ed. John Goldsmith. Chicago: University of Chicago Press.

Goldsmith, John. 2001. The unsupervised learning of natural language morphology. *Computational Linguistics* .

Goldsmith, John. 2007a. Probability for linguists. In *Mathématiques et Sciences Humaine*.

Goldsmith, John. 2007b. Towards a new empiricism. In *Recherches linguistiques a Vincennes*, ed. Joaquim Brandao de Carvalho, volume 36.

Goldsmith, John, and Jason Riggle. 2007. Information theoretical approaches to phonological structure: The case of vowel harmony. Under review.

Goldsmith, John, and Aris Xanthos. 2006. Discovering phonological categories. Under review.

Hare. 1990. The role of similarity in hungarian vowel harmony: a connectionist account. *Connection Science* .

Kirchner, Robert. 1993. Turkish vowel harmony and disharmony: An optimality theoretic account. In *Rutgers Optimality Workshop I (ROW-I)*.

Larson, Gary. 1993. Dyamic computational models and the representation of phonological information. Doctoral Dissertation, University of Chicago.

Manning, Chris, and Hinrich Schütze. 1999. *Foundations of statistical natural language processing*. MIT Press.

de Marken, Carl. 1996. Unsupervised language acquisition. Doctoral Dissertation, MIT.

Mohri, Mehryar. 2004. Weighted finite-state transducer algorithms: An overview. Technical report, AT&T Labs – Research, Shannon Laboratory.

van Oostendorp, Marc. 2005. Greek vowel harmony is no vowel harmony. Unpublished workshop handout.

Pierrehumbert, Janet. 2001. Stochastic phonology. *GLOT* .

Prince, Alan, and Paul Smolensky. 2004. *Optimality theory: Constraint interaction in generative grammar*. Blackwell.

Rabiner, L, and BH Juang. 1993. *Fundamentals of speech recognition*. Prentice-Hall.

Revithiadou, Anthi, Marc van Oostendorp, Kalomoira Nikolou, and Maria anna Tiliopoulou. 2005. Vowel harmony in contact-induced systems: The case of asia minor dialects of greek. Manuscript, University of the Aegean and Meertens Institute.

Riggle, Jason. 2004. Generation recognition and learning in finite state optimality theory. Doctoral Dissertation, UCLA.

Ringen, Catherine O., and Orvokki Heinämäki. 1999. Variation in finnish vowel harmony: An ot account. *Natural Language and Linguistic Theory* .

Rissanen, Jorma. 1978. Modelling by shortest data description. *Automatica* 14:445–471.

Seidenberg, Mark S. 1997. Language acquisition and use: Learning and applying probabilistic constraints. *Science* .

Shannon, Claude, and Warren Weaver. 1949. *The mathematical theory of communication*. Urbana: University of Illinois Press.

Solomonoff, Ray. 1997. The discovery of algorithmic probability. *JCSS* .