# Vowel Harmony Acquisition

Jordan Kodner & Spencer Caplan*

**NECPHON 2016** UMass-Amherst

*Represents equal contribution from both authors

Penn

# Vowel Alternation Patterns

**Turkish Allomorphy:**

- Plural *-lar/-ler*
  - *Baş-**lar*** vs. *Beşev-**ler***

**Fula Stem Alternations:**

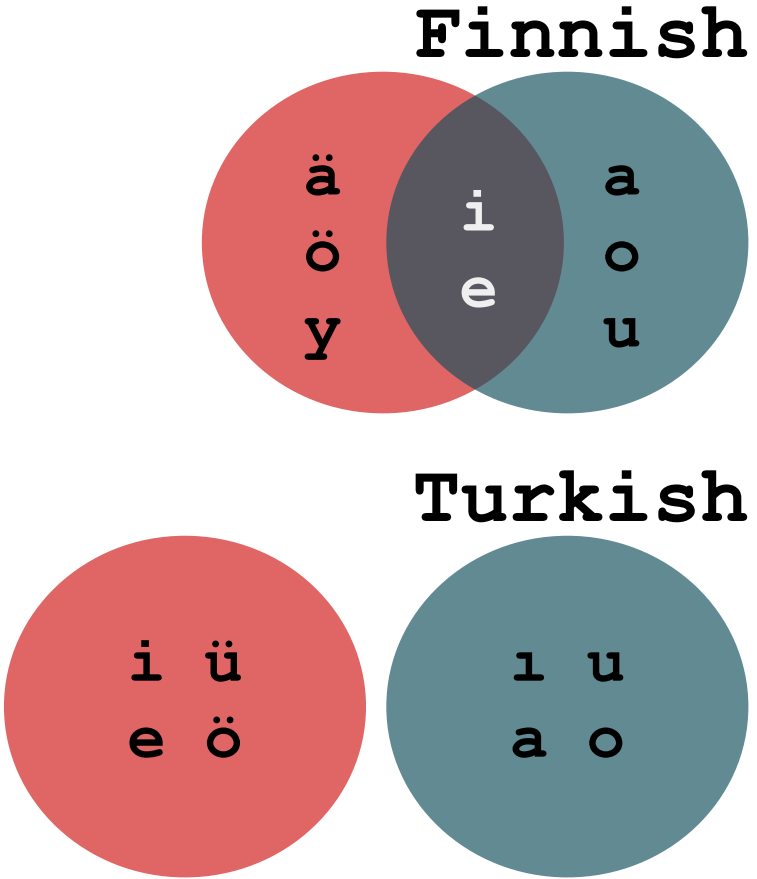
- With *-ɔn* suffix
  - ***mbeel**-u ~ **mbɛɛl**-ɔn* ‘shadow’
  - ***peec**-i ~ **pɛɛc**-ɔn* (proper noun)

**Vowel Alternation in Finnish:**

- *kumarreksituteskenteleentuvaisehkollaismaisekkuudellisenne skenteluttele**matto**mammuuksissan**sakaanko**pa**han*** **vs.**
- *epäjärjestelmällistyttä**mättö**myydellän**säkäänkö**hän***

# What is Vowel Harmony?

- Language-wide vowel alternation patterns
  - Patterns systemic across roots and affixes
  - May or may not affect borrowed vocabulary
- Vowels partitioned into sets
  - Harmonizing classes
  - Neutral vowels
- Caused by *feature spreading*
  - Frontness (**Turkish**, **Finnish**)
  - ATR (Mongolian, Javanese, Fula)
  - Roundness (**Turkish**, **Warlpiri**)



# Quantitative Metrics of Harmony

- Typologies in theoretical phonology are true and useful abstractions of a system
  - Global vs. local
  - Opaque vs. transparent
  - OT Constraints
- But the learner does not have direct access to these
  - direct input is just a surface form!
- Learning model is used to translate between raw input and abstract input/output
- Utility of statistical harmony metrics
  - used to quantify the degree of (dis)harmony present in a language (and thus that a child will be exposed to) (Sanders and Harrison 2012)

# Automatic Harmony Characterization

- Seven-month-olds (with no feedback or annotation) (Mintz et al. 2006)
- So harmony characterization should be easy
- Yet few previous models exist

Sample 6-month-old

**Our Approach**

- Leverage distributional asymmetries in small wordlists (tested on as few as 500 types)

# Previous Models

Goldsmith and Riggle (2012) use an HMM and Boltzmann distribution to model Finnish harmony learning, but with limitations:

- Model does not represent an acquisition pipeline
  - the HMM is able to partition harmonizing vowels only if provided with the set of neutral vowels up front
  - Doesn’t differentiate between input with (e.g. Finnish) and without (e.g. English) vowel harmony present
- No robust psychological motivation for the computational tools employed

# Framework of a Good Model of Harmony Acquisition

- Limited Data
- Pre-segmentation (or mid-segmentation)
  - Running text (phonemes) rather than neatly cut words
  - No frequency counts (need to be able to handle high frequency exceptions)
  - But V/C tiers are accessible! (Newport and Aslin 2004)
- Psychologically motivated tools
  - Any calculations posited should be able to implemented by the learner
- Online processing?
  - Algorithms often assume we have all our data before we try to learn from it; in reality we need to learn from input as it is encountered

# Automatic Harmony Characterization by Distribution

- What the fingerprint of vowel harmony looks like
  - Divergence from a base uniform distribution
- Frequency effects don’t matter
  - Mutual information corrects for the frequency bias of co-occurrence probability
  - Handling issue of marginal vowels
- Differentiating neutral from harmonizing vowels
- Evaluating efficacy of proposed clustering

# Algorithm - on Wordlists

**Input**: Wordlist with frequencies, list of vowels
**Output**: Partitions of vowels into neutral and harmony sets

**Characterize Harmony:**

1. **Trim tail off dataset**
2. Tabulate vowel-vowel cooccurrence matrix
3. Convert counts to mutual information
4. Identify neutral vowels
5. K-means clustering (k=2) of remaining vowels
6. Check that harmony sets partitioned by single feature *F*.
7. Output neutral vowels and harmony sets
8. Collapse on *F* and repeat algorithm

# Algorithm - on Wordlists

**Input**: Wordlist with frequencies, list of vowels
**Output**: Partitions of vowels into neutral and harmony sets

**Characterize Harmony:**

1. Trim tail off dataset
2. **Tabulate vowel-vowel cooccurrence matrix**
3. Convert counts to mutual information
4. Identify neutral vowels
5. K-means clustering (k=2) of remaining vowels
6. Check that harmony sets partitioned by single feature *F*.
7. Output neutral vowels and harmony sets
8. Collapse on *F* and repeat algorithm

**Calculating cooccurrence:**

- Whole-word context
- tier-adjacent context

**Whole-word context toy example:**

```
Corpus:          Coocurrence matrix:
2 aba                      a    e
2 aeb                a     2    4
1 eeba               e     4    1

Vowel frequencies
7 a
4 b
```

# Algorithm - on Wordlists

**Input**: Wordlist with frequencies, list of vowels
**Output**: Partitions of vowels into neutral and harmony sets

**Characterize Harmony:**

1. Trim tail off dataset
2. Tabulate vowel-vowel cooccurrence matrix
3. **Convert counts to mutual information**
4. Identify neutral vowels
5. K-means clustering (k=2) of remaining vowels
6. Check that harmony sets partitioned by single feature *F*.
7. Output neutral vowels and harmony sets
8. Collapse on *F* and repeat algorithm

**Whole-word context toy example:**

```
Corpus:          Coocurrence matrix:
2 aba                      a    e
2 aeb              a       2    4
1 eeba             e       4    1

Vowel frequencies
7 a
4 e

1.  Calculate vowel probabilities
    P(a) = 7/13
    P(b) = 4/13
```

# Algorithm - on Wordlists

**Input**: Wordlist with frequencies, list of vowels
**Output**: Partitions of vowels into neutral and harmony sets

**Characterize Harmony:**

1. Trim tail off dataset
2. Tabulate vowel-vowel cooccurrence matrix
3. **Convert counts to mutual information**
4. Identify neutral vowels
5. K-means clustering (k=2) of remaining vowels
6. Check that harmony sets partitioned by single feature *F*.
7. Output neutral vowels and harmony sets
8. Collapse on *F* and repeat algorithm

**Whole-word context toy example:**

2. **Normalize columns by vowel frequency**

**e.g. norm(a|a) = count(a|a)/P(a)**

**Coocurrence matrix:**

| | a | e |
|---|---|---|
| a | 2/P(a) | 4/P(e) |
| e | 4/P(a) | 1/P(e) |

# Algorithm - on Wordlists

**Input**: Wordlist with frequencies, list of vowels
**Output**: Partitions of vowels into neutral and harmony sets

**Characterize Harmony:**

1. Trim tail off dataset
2. Tabulate vowel-vowel cooccurrence matrix
3. **Convert counts to mutual information**
4. Identify neutral vowels
5. K-means clustering (k=2) of remaining vowels
6. Check that harmony sets partitioned by single feature *F*.
7. Output neutral vowels and harmony sets
8. Collapse on *F* and repeat algorithm

**Whole-word context toy example:**

3. Convert to probabilities by row
   e.g.
   norm(*|a) = norm(a|a) + norm(e|a)
   P(a|a) = norm(a|a) / norm(*|a)

   Coocurrence matrix:

| | a | e |
|---|---|---|
| a | P(a\|a) | P(e\|a) |
| e | P(a\|e) | P(e\|e) |

Final values are probabilities [0,1]

# Algorithm - on Wordlists

**Input**: Wordlist with frequencies, list of vowels
**Output**: Partitions of vowels into neutral and harmony sets

**Characterize Harmony:**

1. Trim tail off dataset
2. Tabulate vowel-vowel cooccurrence matrix
3. Convert counts to mutual information
4. **Identify neutral vowels**
5. K-means clustering (k=2) of remaining vowels
6. Check that harmony sets partitioned by single feature *F*.
7. Output neutral vowels and harmony sets
8. Collapse on *F* and repeat algorithm

**Identifying neutral vowels:**

- Vowels with sufficiently level mutual information
- Threshold proportional to cardinality of vowel set

# Algorithm - on Wordlists

**Input**: Wordlist with frequencies, list of vowels
**Output**: Partitions of vowels into neutral and harmony sets

**Characterize Harmony:**

1. Trim tail off dataset
2. Tabulate vowel-vowel cooccurrence matrix
3. Convert counts to mutual information
4. Identify neutral vowels
5. **K-means clustering (k=2) of remaining vowels**
6. Check that harmony sets partitioned by single feature *F*.
7. Output neutral vowels and harmony sets
8. Collapse on *F* and repeat algorithm

**K-means clustering:**

- Harmony is an opposition between **2** sets
- Cluster on  normalized cooccurence probability vectors

# Algorithm - on Wordlists

**Input**: Wordlist with frequencies, list of vowels
**Output**: Partitions of vowels into neutral and harmony sets

**Characterize Harmony:**

1. Trim tail off dataset
2. Tabulate vowel-vowel cooccurrence matrix
3. Convert counts to mutual information
4. Identify neutral vowels
5. K-means clustering (k=2) of remaining vowels
6. **Check that harmony sets partitioned by single feature *F*.**
7. Output neutral vowels and harmony sets
8. Collapse on *F* and repeat algorithm

**Partitioning on *F*:**

- Harmony is an alternation on a phonological feature
- So proposed harmony must alternate on a feature

# Algorithm - on Wordlists

**Input**: Wordlist with frequencies, list of vowels
**Output**: Partitions of vowels into neutral and harmony sets

**Characterize Harmony:**

1. Trim tail off dataset
2. Tabulate vowel-vowel cooccurrence matrix
3. Convert counts to mutual information
4. Identify neutral vowels
5. K-means clustering (k=2) of remaining vowels
6. Check that harmony sets partitioned by single feature *F*.
7. **Output neutral vowels and harmony sets**
8. Collapse on *F* and repeat algorithm

# Algorithm - on Wordlists

**Input**: Wordlist with frequencies, list of vowels
**Output**: Partitions of vowels into neutral and harmony sets

**Characterize Harmony:**

1. Trim tail off dataset
2. Tabulate vowel-vowel cooccurrence matrix
3. Convert counts to mutual information
4. Identify neutral vowels
5. K-means clustering (k=2) of remaining vowels
6. Check that harmony sets partitioned by single feature *F*.
7. Output neutral vowels and harmony sets
8. **Collapse on *F* and repeat algorithm**

**Secondary Harmony:**

- Collapsing vowels on F removes primary harmony signal
- Rerunning the algorithm discovers secondary harmony

# Extending to Unsegmented Utterances

- Children as young as seven months (Mintz et al. 2006) have been shown to be sensitive to vowel harmony
- Harmony sensitivity is language dependent (Kabak et al. 2010)
- Speakers can recognize words on the basis of harmony cues (Suomi et al., 1997; Vroomen et al. 1998)

**Our Approach**

- Test the algorithm on unsegmented utterances
- Use Wikipedia sentences with spaces removed

**Challenge:** Cooccurance across actual word boundaries introduces noise

# Example Utterances

Finnish Examples (Wikipedia):

- **Amsterdaminpaikallerakennettiinensimmäisetpuutalotluvunalkupuolella**
- **Näinasukkaatsaivattästäkoituneettulotitselleen**
- **kauppalaajenijajaluvuillakaupunginasukaslukukasvoinopeasti**

Warlpiri Examples (Steve Swartz):

- **Jajarnumayinkili**
- **Ngulajangkajupakarninjawarnurlujumardalukinkinkujurnuwiyi**
- **jaantakupinyikangamiturakikirlangu**

# Algorithm - on Utterances

**Input**: Utterances with no segmentation, list of vowels
**Output**: Partitions of vowels into neutral and harmony sets

**Characterize Harmony:**

1. Trim tail off dataset
2. Tabulate vowel-vowel cooccurrence matrix
3. Convert counts to mutual information
4. Identify neutral vowels
5. K-means clustering (k=2) of remaining vowels
6. Check that harmony sets partitioned by single feature *F*.
7. Output neutral vowels and harmony sets
8. Collapse on *F* and repeat algorithm

# Algorithm - on Utterances

**Input**: Utterances with no segmentation, list of vowels
**Output**: Partitions of vowels into neutral and harmony sets

**Characterize Harmony:**

1. ~~**Trim tail off dataset**~~
2. Tabulate vowel-vowel cooccurrence matrix
3. Convert counts to mutual information
4. Identify neutral vowels
5. K-means clustering (k=2) of remaining vowels
6. Check that harmony sets partitioned by single feature *F*.
7. Output neutral vowels and harmony sets
8. Collapse on *F* and repeat algorithm

# Algorithm - on Utterances

**Input**: Utterances with no segmentation, list of vowels
**Output**: Partitions of vowels into neutral and harmony sets

**Characterize Harmony:**

1. **Tabulate vowel-vowel cooccurrence matrix** ← **Tier-adjacent within *k* (we used *k*=1)**
2. Convert counts to mutual information
3. Identify neutral vowels
4. K-means clustering (k=2) of remaining vowels
5. Check that harmony sets partitioned by single feature *F*.
6. Output neutral vowels and harmony sets
7. Collapse on *F* and repeat algorithm

# Results on Utterances

| Language | Primary Harmony? | %V Correct | Secondary Harmony? | %V Correct |
|---|---|---|---|---|
| **Turkish** | ☑ | 100% | ☑ | 100% |
| **Finnish** | ☑ | 100% | ☒ | 100% |
| **Warlpiri** | ☑ | 100% | ☒ | 100% |
| **English** | ☒ | 100% | ☒ | 100% |

# Ongoing and Future Work

- Ongoing
  - Capturing secondary harmony processes
  - Differentiating global from local, opaque vs. transparent neutral vowels
- Future
  - Differentiating non-productive disharmonic data from productive harmony process (e.g. Estonian vs. Finnish)
  - Online processing (given an incoming stream of data)
  - How much segmentation is possible given only output of harmony model