# Efficient Learning of Tier-based Strictly *k*-Local languages

Adam Jardine
Rutgers University

Kevin McMullin
University of Ottawa

RUTGERS uOttawa

The 11th International Conference on Language and Automata Theory and Applications
Umeå University, Umeå, Sweden
March 6, 2017

# Overview

- The Tier-based Strictly $k$-Local ($TSL_k$) languages are formal languages where dependencies hold independent of some set of 'ignored' symbols
  - $TSL_k$ argued to be a close approximation of attested linguistic sound patterns
- We introduce the Tier-based $k$-Strictly Local Inference Algorithm ($k$TSLIA)
- Identifies $TSL_k$ languages in quadratic time; size of sample necessary for identification is bounded by a constant
- We do this by proving new properties about TSL languages that allow the learner to discover which symbols can(not) be ignored

**Part 1 (of 2):**

- Introduce and motivate $TSL_k$ languages
- Identify learning paradigm

# Some Notation

- $\Sigma$ is alphabet; $\rtimes, \ltimes \notin \Sigma$ are **boundary symbols**
- For $w \in \Sigma^*$, $u$ is a $k$-**factor** of $w$ if $\rtimes w \ltimes = v_1 u v_2$ and $|u| = k$.

$$\mathtt{fac}_k(w) \stackrel{\text{def}}{=} \begin{cases} \{u \mid u \text{ is a } k\text{-factor of } \rtimes w \ltimes\} & \text{if } |\rtimes w \ltimes| > k \\ \{\rtimes w \ltimes\} & \text{otherwise} \end{cases}$$

# Some Notation

- $\Sigma$ is alphabet; $\rtimes, \ltimes \notin \Sigma$ are **boundary symbols**
- For $w \in \Sigma^*$, $u$ is a $k$-**factor** of $w$ if $\rtimes w \ltimes = v_1 u v_2$ and $|u| = k$.

$$\texttt{fac}_k(w) \stackrel{\text{def}}{=} \begin{cases} \{u \mid u \text{ is a } k\text{-factor of } \rtimes w \ltimes\} & \text{if } |\rtimes w \ltimes| > k \\ \{\rtimes w \ltimes\} & \text{otherwise} \end{cases}$$

- $\texttt{fac}_3(abbba) = \{\rtimes ab, abb, bbb, bba, ba\ltimes\}$

# Some Notation

- $\Sigma$ is alphabet; $\rtimes, \ltimes \notin \Sigma$ are **boundary symbols**
- For $w \in \Sigma^*$, $u$ is a ***k*-factor** of $w$ if $\rtimes w \ltimes = v_1 u v_2$ and $|u| = k$.

$$\texttt{fac}_k(w) \stackrel{\text{def}}{=} \begin{cases} \{u \mid u \text{ is a } k\text{-factor of } \rtimes w \ltimes\} & \text{if } |\rtimes w \ltimes| > k \\ \{\rtimes w \ltimes\} & \text{otherwise} \end{cases}$$

- $\texttt{fac}_3(abbba) = \{\rtimes ab, abb, bbb, bba, ba\ltimes\}$
- Extends straightforwardly to $\texttt{fac}_k(L)$ for set $L \subseteq \Sigma^*$
- $\texttt{fac}_k(L)$ computed in time linear in $||L||$

# The Strictly $k$-Local Languages

- The **Strictly $k$-Local ($\mathbf{SL}_k$)** languages [MP71, RHF$^+$13] model 'local' dependencies

$$R \subseteq \texttt{fac}_k(\Sigma^*)$$

- The language is the set of strings that contain no banned $k$-factors

$$L(R) \stackrel{\text{def}}{=} \{w \in \Sigma^* \mid \texttt{fac}_k(w) \cap R = \emptyset\}$$

# The Strictly $k$-Local Languages

- The **Strictly $k$-Local ($\mathbf{SL}_k$)** languages [MP71, RHF$^+$13] model 'local' dependencies

$$R \subseteq \texttt{fac}_k(\Sigma^*)$$

- The language is the set of strings that contain no banned $k$-factors

$$L(R) \stackrel{\text{def}}{=} \{w \in \Sigma^* \mid \texttt{fac}_k(w) \cap R = \emptyset\}$$

- $R = \{\rtimes ab\}$; $abbba \notin L(R)$, $babab \in L(R)$

# The Tier-based Strictly $k$-Local Languages

- The $\mathrm{TSL}_k$ languages [HRT11] generalize $\mathrm{SL}_k$ languages with a **tier** $T \subseteq \Sigma$ over which $R$ is evaluated
- All symbols in $\Sigma - T$ ignored

$$\Sigma = \{a, b\}, T = \{b\}, R = \{bbb\}$$

$$abbba \quad , \quad abbaaaba \quad , \quad abaaaba$$

# The Tier-based Strictly $k$-Local Languages

- The $\text{TSL}_k$ languages [HRT11] generalize $\text{SL}_k$ languages with a **tier** $T \subseteq \Sigma$ over which $R$ is evaluated
- All symbols in $\Sigma - T$ ignored

$$\Sigma = \{a, b\}, T = \{b\}, R = \{bbb\}$$

$$abbba \quad , \quad abbaaaba \quad , \quad abaaaba$$

# The Tier-based Strictly $k$-Local Languages

- The $\mathrm{TSL}_k$ languages [HRT11] generalize $\mathrm{SL}_k$ languages with a **tier** $T \subseteq \Sigma$ over which $R$ is evaluated
- All symbols in $\Sigma - T$ ignored

$$\Sigma = \{a, b\}, T = \{b\}, R = \{bbb\}$$

$$abbba \notin L, \quad abbaaaba \notin L, \quad abaaaba \in L$$

# The Tier-based Strictly $k$-Local Languages

- More formally, $\mathrm{TSL}_k$ grammar is $G = \langle T, R \subseteq \mathtt{fac}_k(T^*) \rangle$

$$\mathtt{erase}_T(w) \stackrel{\text{def}}{=} \begin{cases} \mathtt{erase}_T(u) \cdot \sigma & \text{if } w = u\sigma,\ u \in \Sigma^*, \sigma \in T \\ \mathtt{erase}_T(u) & \text{if } w = u\sigma,\ u \in \Sigma^*, \sigma \notin T \end{cases}$$

# The Tier-based Strictly $k$-Local Languages

- More formally, $\mathrm{TSL}_k$ grammar is $G = \langle T, R \subseteq \mathtt{fac}_k(T^*) \rangle$

$$\mathtt{erase}_T(w) \stackrel{\text{def}}{=} \begin{cases} \mathtt{erase}_T(u) \cdot \sigma & \text{if } w = u\sigma,\ u \in \Sigma^*, \sigma \in T \\ \mathtt{erase}_T(u) & \text{if } w = u\sigma,\ u \in \Sigma^*, \sigma \notin T \end{cases}$$

- If $\Sigma = \{a, b\}, T = \{b\}$, $\mathtt{erase}_T(abbaaaba) = bbb$
- The language is the set of strings that contain no banned $k$-factors after erasing all non-tier symbols

$$L(G) \stackrel{\text{def}}{=} \{w \mid \mathtt{fac}_k(\mathtt{erase}_T(w)) \cap R = \emptyset\}$$

# Linguistic relevance of $SL_k$ and $TSL_k$

- $SL_k$ and $TSL_k$ languages nontrivially model **phonotactics**; speakers' knowledge of how sounds are used to form words in their language [Hei10, Hei11, HRT11]
- English $= \{$shrimp, blink, bork, flump, ...$\}$
- sr $\in R_{\text{English}}$ (srimp, srit, ...$\notin$ English)

# Linguistic relevance of $\mathrm{SL}_k$ and $\mathrm{TSL}_k$

- Finnish [Nev10, Odd94]

  pöütä-nä ‘table-ESS’ ulko-ta ‘outside-ABL’

  väkkärä-nä ‘pinwheel-ESS’ pappi-na ‘priest-ESS’

# Linguistic relevance of $\mathrm{SL}_k$ and $\mathrm{TSL}_k$

- Finnish [Nev10, Odd94]

  pöütä-nä 'table-ESS' ulko-ta 'outside-ABL'

  väkkärä-nä 'pinwheel-ESS' pappi-na 'priest-ESS'
- $T = \{$ö,ü,ä,o,u,a$\}$ (notice no $\{i, e\}$!)

# Linguistic relevance of $SL_k$ and $TSL_k$

- Finnish [Nev10, Odd94]

  pöütä-nä 'table-ESS' ulko-ta 'outside-ABL'

  väkkärä-nä 'pinwheel-ESS' pappi-na 'priest-ESS'
- $T = \{$ö,ü,ä,o,u,a$\}$ (notice no $\{i, e\}$!)
- äa$\in R_{\text{Finnish}}$: päppi-na $\notin$ Finnish

# Linguistic relevance of $SL_k$ and $TSL_k$

- Tiers are language-specific:

| | | |
|---|---|---|
| Turkish: | Vowels | [CS82] |
| Finnish: | Vowels except {i, e} | [Rin75] |
| Sundanese: | {l, r} | [Coh92] |
| Latin: | {l, r, m, g} | [Cse10] |
| Samala: | {s, ʃ} | [RW04] |
| Koorete: | {s, ʃ, b, r, g, d} | [MH16] |

# Learning goal

- For a given $\Sigma$ and $k$ the set of grammars $\langle T, R \rangle$ is finite
- Thus learnable via enumeration [Gol67]
- Is there a smarter, efficient learner?

# Learning paradigm

- 'Efficient learning' means **exact identification in the limit in polynomial time and data** [dlH97]
- A **characteristic sample** $I_C$ for a language $L$ for an algorithm $A$ is a finite set such that for all $I \supseteq I_C$ of $L$, $L = L(A(I))$
- Goal is $A$ that
  - identifies $L$ if $I$ contains $I_C$ for $L$
  - runs in time polynomial in $||I||$ for any input $I$
  - $||I_C||$ for any TSL$_k$ language $L$ is polynomial in the size of its grammar

# Learning paradigm

- Such an $A$ exists for $\mathrm{TSL}_2$ which runs in $||I||^4$ time [JH16]
- We present an $A$ for any $k$ which runs in $||I||^2$ time

**Part 2 (of 2):**

- Define canonical $\text{TSL}_k$ grammar
- Show two properties of $T$ and $\Sigma - T$ for canonical grammar
- Show how algorithm learns using these properties

# Canonical TSL$_k$ grammar

**Definition (Canonical $TSL_k$ grammar)**

A $TSL_k$ grammar $G = \langle T, R \rangle$ is *canonical* iff for any $TSL_k$ grammar $G' = \langle T', R' \rangle$, $L(G) = L(G')$ and $G \neq G'$ implies $T \subset T'$.

# Canonical $\mathrm{TSL}_k$ grammar

**Definition (Canonical $TSL_k$ grammar)**

A $TSL_k$ grammar $G = \langle T, R \rangle$ is *canonical* iff for any $TSL_k$ grammar $G' = \langle T', R' \rangle$, $L(G) = L(G')$ and $G \neq G'$ implies $T \subset T'$.

- $\Sigma = \{a, b\}$

$$G_1 = \langle T_1 = \{a, b\}, R_1 = \{\rtimes bb, bbb, bb\ltimes, abb, bba, bab\} \rangle$$
$$G_2 = \langle T_2 = \{b\}, \quad R_2 = \{\rtimes bb, bbb, bb\ltimes\} \rangle$$

$$L(G_1) = L(G_2) = \{\lambda, a, aa, ba, ab, aaa, aab, aba, baa, ...\}$$

# Canonical TSL$_k$ grammar

**Definition (Canonical $TSL_k$ grammar)**

A $TSL_k$ grammar $G = \langle T, R \rangle$ is *canonical* iff for any $TSL_k$ grammar $G' = \langle T', R' \rangle$, $L(G) = L(G')$ and $G \neq G'$ implies $T \subset T'$.

- $\Sigma = \{a, b\}$

$$G_1 = \langle T_1 = \{a, b\}, R_1 = \{\rtimes bb, bbb, bb\ltimes, abb, bba, bab\} \rangle$$
$$G_2 = \langle T_2 = \{b\}, \quad R_2 = \{\rtimes bb, bbb, bb\ltimes\} \rangle$$

$$L(G_1) = L(G_2) = \{\lambda, a, aa, ba, ab, aaa, aab, aba, baa, ...\}$$

# Properties of canonical grammar

Lemma (The '*R* tier member lemma')

*If $G = \langle T, R \rangle$ is a canonical TSL$_k$ grammar, then for all $\sigma \in T$ which appear in R, there is at least one $v_1\sigma v_2 \in R$ such that $v_1 v_2 \in \texttt{fac}_{k-1}(L(G))$.*

- Intuition: Otherwise, $\sigma$ plays no role in determining language

# Properties of canonical grammar

## Lemma (The '*R* tier member lemma')

*If* $G = \langle T, R \rangle$ *is a canonical* $TSL_k$ *grammar, then for all* $\sigma \in T$ *which appear in* $R$*, there is at least one* $v_1 \sigma v_2 \in R$ *such that* $v_1 v_2 \in \mathtt{fac}_{k-1}(L(G))$.

- Intuition: Otherwise, $\sigma$ plays no role in determining language
- Example:
  $G = \langle T = \{a, b\}, R = \{bab\} \rangle$

# Properties of canonical grammar

## Lemma (The '*R* tier member lemma')

*If* $G = \langle T, R\rangle$ *is a canonical* $TSL_k$ *grammar, then for all* $\sigma \in T$ *which appear in R, there is at least one* $v_1\sigma v_2 \in R$ *such that* $v_1 v_2 \in \mathtt{fac}_{k-1}(L(G))$.

- Intuition: Otherwise, $\sigma$ plays no role in determining language
- Example:
  $G = \langle T = \{a, b\}, R = \{bab\}\rangle$
  $ababa \notin L(G)$, but $abba \in L(G)$

# Properties of canonical grammar

Lemma (The '$R$ tier member lemma')

*If $G = \langle T, R\rangle$ is a canonical $TSL_k$ grammar, then for all $\sigma \in T$ which appear in $R$, there is at least one $v_1\sigma v_2 \in R$ such that $v_1 v_2 \in \mathtt{fac}_{k-1}(L(G))$.*

- Intuition: Otherwise, $\sigma$ plays no role in determining language
- Example:
  $G = \langle T = \{a, b\}, R = \{bab\}\rangle$
  $ababa \notin L(G)$, but $abba \in L(G)$

  $G' = \langle T = \{a, b\}, R' = \{\rtimes bb, bbb, bb\ltimes, abb, bba, bab\}\rangle$

# Properties of canonical grammar

## Lemma (The '*R* tier member lemma')

*If* $G = \langle T, R \rangle$ *is a canonical* $TSL_k$ *grammar, then for all* $\sigma \in T$ *which appear in R, there is at least one* $v_1 \sigma v_2 \in R$ *such that* $v_1 v_2 \in \mathtt{fac}_{k-1}(L(G))$.

- Intuition: Otherwise, $\sigma$ plays no role in determining language
- Example:
  $G = \langle T = \{a, b\}, R = \{bab\} \rangle$
  $ababa \notin L(G)$, but $abba \in L(G)$

  $G' = \langle T = \{a, b\}, R' = \{\rtimes bb, bbb, bb\ltimes, abb, bba, bab\} \rangle$
  $ababa, abba \notin L(G')$

# Properties of canonical grammar

**Lemma (The '$R$ tier member lemma')**

*If $G = \langle T, R \rangle$ is a canonical $TSL_k$ grammar, then for all $\sigma \in T$ which appear in $R$, there is at least one $v_1 \sigma v_2 \in R$ such that $v_1 v_2 \in \mathtt{fac}_{k-1}(L(G))$.*

- Intuition: Otherwise, $\sigma$ plays no role in determining language
- Example:

  $G = \langle T = \{a, b\}, R = \{bab\} \rangle$

  $ababa \notin L(G)$, but $abba \in L(G)$

  $G' = \langle T = \{a, b\}, R' = \{\rtimes bb, bbb, bb\ltimes, abb, bba, bab\} \rangle$

  $ababa, abba \notin L(G')$

  $G'' = \langle T' = \{b\}, R'' = \{\rtimes bb, bbb, bb\ltimes\} \rangle$

  $L(G') = L(G'')!$

# Properties of canonical grammar

Lemma (The 'non-tier member lemma')

*For a canonical $TSL_k$ grammar $G$, the following hold iff $\sigma \notin T$:*

a. $\forall v_1 v_2 \in \mathtt{fac}_{k-1}(L(G)), v_1 \sigma v_2 \in \mathtt{fac}_k(L(G))$

# Properties of canonical grammar

Lemma (The 'non-tier member lemma')

*For a canonical* $TSL_k$ *grammar* $G$*, the following hold iff* $\sigma \notin T$*:*

a. $\forall v_1 v_2 \in \mathtt{fac}_{k-1}(L(G)), v_1 \sigma v_2 \in \mathtt{fac}_k(L(G))$

- $\Sigma = \{a, b, c\}, G = \langle T = \{b, c\}, R = \{bbb\} \rangle$

# Properties of canonical grammar

Lemma (The 'non-tier member lemma')

*For a canonical $TSL_k$ grammar $G$, the following hold iff $\sigma \notin T$:*

a. $\forall v_1 v_2 \in \mathtt{fac}_{k-1}(L(G)), v_1 \sigma v_2 \in \mathtt{fac}_k(L(G))$

- $\Sigma = \{a, b, c\}, G = \langle T = \{b, c\}, R = \{bbb\} \rangle$
  $abba, ababa \in L(G)$

# Properties of canonical grammar

Lemma (The 'non-tier member lemma')

*For a canonical $TSL_k$ grammar $G$, the following hold iff $\sigma \notin T$:*

a. $\forall v_1 v_2 \in \mathtt{fac}_{k-1}(L(G)), v_1 \sigma v_2 \in \mathtt{fac}_k(L(G))$

- $\Sigma = \{a, b, c\}, G = \langle T = \{b, c\}, R = \{bbb\} \rangle$
  $abba, ababa \in L(G)$ , $abbba \notin L(G)$,

# Properties of canonical grammar

Lemma (The 'non-tier member lemma')

*For a canonical* $TSL_k$ *grammar* $G$*, the following hold iff* $\sigma \notin T$*:*

a. $\forall v_1 v_2 \in \mathtt{fac}_{k-1}(L(G)), v_1 \sigma v_2 \in \mathtt{fac}_k(L(G))$

- $\Sigma = \{a, b, c\}, G = \langle T = \{b, c\}, R = \{bbb\} \rangle$
  $abba, ababa \in L(G)$ , $abbba \notin L(G)$, $abba, abcba \in L(G)$

# Properties of canonical grammar

Lemma (The 'non-tier member lemma')

*For a canonical $TSL_k$ grammar $G$, the following hold iff $\sigma \notin T$:*

a. $\forall v_1 v_2 \in \mathtt{fac}_{k-1}(L(G)), v_1 \sigma v_2 \in \mathtt{fac}_k(L(G))$

b. $\forall v_1 \sigma v_2 \in \mathtt{fac}_{k+1}(L(G)), v_1 v_2 \in \mathtt{fac}_k(L(G))$

- $\Sigma = \{a, b, c\}, G = \langle T = \{b, c\}, R = \{bbb\} \rangle$
  $abba, ababa \in L(G)$ , $abbba \notin L(G)$, $abba, abcba \in L(G)$

# Properties of canonical grammar

Lemma (The 'non-tier member lemma')

*For a canonical $TSL_k$ grammar G, the following hold iff $\sigma \notin T$:*

a. $\forall v_1 v_2 \in \mathtt{fac}_{k-1}(L(G)), v_1 \sigma v_2 \in \mathtt{fac}_k(L(G))$

b. $\forall v_1 \sigma v_2 \in \mathtt{fac}_{k+1}(L(G)), v_1 v_2 \in \mathtt{fac}_k(L(G))$

- $\Sigma = \{a, b, c\}, G = \langle T = \{b, c\}, R = \{bbb\}\rangle$
  $abba, ababa \in L(G)$ , $abbba \notin L(G)$, $abba, abcba \in L(G)$
  $abbcba \in L(G), abbba \notin L(G)$

# The algorithm

**The non-tier member lemma:** Iff $\sigma \notin T$:

a. $\forall v_1 v_2 \in \texttt{fac}_{k-1}(L(G)), v_1 \sigma v_2 \in \texttt{fac}_k(L(G))$

b. $\forall v_1 \sigma v_2 \in \texttt{fac}_{k+1}(L(G)), v_1 v_2 \in \texttt{fac}_k(L(G))$

$$\Sigma = \{a, b, c\},\ G = \langle T = \{b, c\}, R = \{bbb\} \rangle$$

- The non-tier member lemma uniquely identifies non-tier members

# The algorithm

**The $k$TSLIA:** $\sigma$ from $T$ hypothesis for which

a. $\forall v_1 v_2 \in \texttt{fac}_{k-1}(I), v_1 \sigma v_2 \in \texttt{fac}_k(I)$

b. $\forall v_1 \sigma v_2 \in \texttt{fac}_{k+1}(I), v_1 v_2 \in \texttt{fac}_k(I)$

- Given $I$, the Tier-based Strictly $k$-Local Induction Algorithm ($k$TSLIA) searches through $\texttt{fac}_{k-1}(I)$, $\texttt{fac}_k(I)$, $\texttt{fac}_{k+1}(I)$

# The algorithm

**The $k$TSLIA:** $\sigma$ from $T$ hypothesis for which

a. $\forall v_1 v_2 \in \texttt{fac}_{k-1}(I), v_1 \sigma v_2 \in \texttt{fac}_k(I)$
b. $\forall v_1 \sigma v_2 \in \texttt{fac}_{k+1}(I), v_1 v_2 \in \texttt{fac}_k(I)$

- Given $I$, the Tier-based Strictly $k$-Local Induction Algorithm ($k$TSLIA) searches through $\texttt{fac}_{k-1}(I)$, $\texttt{fac}_k(I)$, $\texttt{fac}_{k+1}(I)$
- Any $\sigma \in \Sigma$ that satisfies both (a) and (b) removed from from hypothesis for $T$

# The algorithm

**The $k$TSLIA:** $\sigma$ from $T$ hypothesis for which

a. $\forall v_1 v_2 \in \texttt{fac}_{k-1}(I), v_1 \sigma v_2 \in \texttt{fac}_k(I)$
b. $\forall v_1 \sigma v_2 \in \texttt{fac}_{k+1}(I), v_1 v_2 \in \texttt{fac}_k(I)$

- Given $I$, the Tier-based Strictly $k$-Local Induction Algorithm ($k$TSLIA) searches through $\texttt{fac}_{k-1}(I)$, $\texttt{fac}_k(I)$, $\texttt{fac}_{k+1}(I)$
- Any $\sigma \in \Sigma$ that satisfies both (a) and (b) removed from from hypothesis for $T$
- Hypothesis for $R$ set to all remaining $\texttt{fac}_k(T^*)$ not in $\texttt{fac}_k(I)$

# The algorithm (example)

**Target:** $G_* = \langle T_* = \{b\}, R_* = \{bbb\}\rangle, \Sigma = \{a, b\}, k = 3$

# The algorithm (example)

**Target:** $G_* = \langle T_* = \{b\}, R_* = \{bbb\}\rangle, \Sigma = \{a, b\}, k = 3$
**Sample:** $\{\lambda, a, b, bb, aaa, bab, abba, aabaabaa\}$

# The algorithm (example)

**Target:** $G_* = \langle T_* = \{b\}, R_* = \{bbb\}\rangle, \Sigma = \{a, b\}, k = 3$
**Sample:** $\{\lambda, a, b, bb, aaa, bab, abba, aabaabaa\}$

| String | 2-factors | 3-factors | 4-factors |
|---|---|---|---|

# The algorithm (example)

**Target:** $G_* = \langle T_* = \{b\}, R_* = \{bbb\}\rangle, \Sigma = \{a, b\}, k = 3$
**Sample:** $\{\lambda, a, b, bb, aaa, bab, abba, aabaabaa\}$

| String | 2-factors | 3-factors | 4-factors |
|---|---|---|---|
| $\lambda$ | $\rtimes \ltimes$ | | |

# The algorithm (example)

**Target:** $G_* = \langle T_* = \{b\}, R_* = \{bbb\}\rangle, \Sigma = \{a, b\}, k = 3$
**Sample:** $\{\lambda, a, b, bb, aaa, bab, abba, aabaabaa\}$

| String | 2-factors | 3-factors | 4-factors |
|---|---|---|---|
| $\lambda$ | $\rtimes\ \ltimes$ | | |
| $a$ | $\rtimes a, a\ltimes$ | $\rtimes a \ltimes$ | |

# The algorithm (example)

**Target:** $G_* = \langle T_* = \{b\}, R_* = \{bbb\}\rangle, \Sigma = \{a, b\}, k = 3$
**Sample:** $\{\lambda, a, b, bb, aaa, bab, abba, aabaabaa\}$

| String | 2-factors | 3-factors | 4-factors |
|---|---|---|---|
| $\lambda$ | $\rtimes \ \ltimes$ | | |
| $a$ | $\rtimes a, a\ltimes$ | $\rtimes a \ltimes$ | |
| $b$ | $\rtimes b, b\ltimes$ | $\rtimes b \ltimes$ | |

# The algorithm (example)

**Target:** $G_* = \langle T_* = \{b\}, R_* = \{bbb\}\rangle, \Sigma = \{a, b\}, k = 3$
**Sample:** $\{\lambda, a, b, bb, aaa, bab, abba, aabaabaa\}$

| String | 2-factors | 3-factors | 4-factors |
|---|---|---|---|
| $\lambda$ | $\rtimes\ \ltimes$ | | |
| $a$ | $\rtimes a, a\ltimes$ | $\rtimes a \ltimes$ | |
| $b$ | $\rtimes b, b\ltimes$ | $\rtimes b \ltimes$ | |
| $bb$ | $bb$ | $\rtimes bb, bb\ltimes$ | $\rtimes bb \ltimes$ |

# The algorithm (example)

**Target:** $G_* = \langle T_* = \{b\}, R_* = \{bbb\}\rangle, \Sigma = \{a, b\}, k = 3$
**Sample:** $\{\lambda, a, b, bb, aaa, bab, abba, aabaabaa\}$

| String | 2-factors | 3-factors | 4-factors |
|---|---|---|---|
| $\lambda$ | $\rtimes\ \ltimes$ | | |
| $a$ | $\rtimes a, a\ltimes$ | $\rtimes a\ltimes$ | |
| $b$ | $\rtimes b, b\ltimes$ | $\rtimes b\ltimes$ | |
| $bb$ | $bb$ | $\rtimes bb, bb\ltimes$ | $\rtimes bb\ltimes$ |
| $aaa$ | $aa$ | $\rtimes aa, aaa, aa\ltimes$ | $\rtimes aaa, aaa\ltimes$ |

# The algorithm (example)

**Target:** $G_* = \langle T_* = \{b\}, R_* = \{bbb\}\rangle, \Sigma = \{a, b\}, k = 3$
**Sample:** $\{\lambda, a, b, bb, aaa, bab, abba, aabaabaa\}$

| String | 2-factors | 3-factors | 4-factors |
|---|---|---|---|
| $\lambda$ | $\rtimes\ \ltimes$ | | |
| $a$ | $\rtimes a, a\ltimes$ | $\rtimes a\ltimes$ | |
| $b$ | $\rtimes b, b\ltimes$ | $\rtimes b\ltimes$ | |
| $bb$ | $bb$ | $\rtimes bb, bb\ltimes$ | $\rtimes bb\ltimes$ |
| $aaa$ | $aa$ | $\rtimes aa, aaa, aa\ltimes$ | $\rtimes aaa, aaa\ltimes$ |
| $bab$ | $ba, ab$ | $\rtimes ba, bab, ab\ltimes$ | $\rtimes bab, bab\ltimes$ |

# The algorithm (example)

**Target:** $G_* = \langle T_* = \{b\}, R_* = \{bbb\}\rangle, \Sigma = \{a, b\}, k = 3$
**Sample:** $\{\lambda, a, b, bb, aaa, bab, abba, aabaabaa\}$

| String | 2-factors | 3-factors | 4-factors |
|---|---|---|---|
| $\lambda$ | $\rtimes\ \ltimes$ | | |
| $a$ | $\rtimes a, a\ltimes$ | $\rtimes a\ltimes$ | |
| $b$ | $\rtimes b, b\ltimes$ | $\rtimes b\ltimes$ | |
| $bb$ | $bb$ | $\rtimes bb, bb\ltimes$ | $\rtimes bb\ltimes$ |
| $aaa$ | $aa$ | $\rtimes aa, aaa, aa\ltimes$ | $\rtimes aaa, aaa\ltimes$ |
| $bab$ | $ba, ab$ | $\rtimes ba, bab, ab\ltimes$ | $\rtimes bab, bab\ltimes$ |
| $abba$ | — | $\rtimes ab, abb, bba, ba\ltimes$ | $\rtimes abb, abba, bba\ltimes$ |

# The algorithm (example)

**Target:** $G_* = \langle T_* = \{b\}, R_* = \{bbb\}\rangle, \Sigma = \{a, b\}, k = 3$
**Sample:** $\{\lambda, a, b, bb, aaa, bab, abba, aabaabaa\}$

| String | 2-factors | 3-factors | 4-factors |
|---|---|---|---|
| $\lambda$ | $\rtimes\ \ltimes$ | | |
| $a$ | $\rtimes a, a\ltimes$ | $\rtimes a\ltimes$ | |
| $b$ | $\rtimes b, b\ltimes$ | $\rtimes b\ltimes$ | |
| $bb$ | $bb$ | $\rtimes bb, bb\ltimes$ | $\rtimes bb\ltimes$ |
| $aaa$ | $aa$ | $\rtimes aa, aaa, aa\ltimes$ | $\rtimes aaa, aaa\ltimes$ |
| $bab$ | $ba, ab$ | $\rtimes ba, bab, ab\ltimes$ | $\rtimes bab, bab\ltimes$ |
| $abba$ | — | $\rtimes ab, abb, bba, ba\ltimes$ | $\rtimes abb, abba, bba\ltimes$ |
| $aabaabaa$ | — | $aab, aba, baa$ | $\rtimes aab, aaba, abaa, baa\ltimes$ |

# The algorithm (example)

**Target:** $G_* = \langle T_* = \{b\}, R_* = \{bbb\} \rangle, \Sigma = \{a, b\}, k = 3$
**Sample:** $\{\lambda, a, b, bb, aaa, bab, abba, aabaabaa\}$

| String | 2-factors | 3-factors | 4-factors |
|---|---|---|---|
| $\lambda$ | $\rtimes \ltimes$ | | |
| $a$ | $\rtimes a, a\ltimes$ | $\rtimes a\ltimes$ | |
| $b$ | $\rtimes b, b\ltimes$ | $\rtimes b\ltimes$ | |
| $bb$ | $bb$ | $\rtimes bb, bb\ltimes$ | $\rtimes bb\ltimes$ |
| $aaa$ | $aa$ | $\rtimes aa, aaa, aa\ltimes$ | $\rtimes aaa, aaa\ltimes$ |
| $bab$ | $ba, ab$ | $\rtimes ba, bab, ab\ltimes$ | $\rtimes bab, bab\ltimes$ |
| $abba$ | — | $\rtimes ab, abb, bba, ba\ltimes$ | $\rtimes abb, abba, bba\ltimes$ |
| $aabaabaa$ | — | $aab, aba, baa$ | $\rtimes aab, aaba,$ $abaa, baa\ltimes$ |

a. $\forall v_1 v_2 \in \mathtt{fac}_{k-1}(I), v_1 \sigma v_2 \in \mathtt{fac}_k(I)$

b. $\forall v_1 \sigma v_2 \in \mathtt{fac}_{k+1}(I), v_1 v_2 \in \mathtt{fac}_k(I)$

# The algorithm (example)

**Target:** $G_* = \langle T_* = \{b\}, R_* = \{bbb\}\rangle, \Sigma = \{a, b\}, k = 3$
**Sample:** $\{\lambda, a, b, bb, aaa, bab, abba, aabaabaa\}$

| String | 2-factors | 3-factors | 4-factors |
|---|---|---|---|
| $\lambda$ | $\rtimes\ \ltimes$ | | |
| $a$ | $\rtimes a, a\ltimes$ | $\rtimes a\ltimes$ | |
| $b$ | $\rtimes b, b\ltimes$ | $\rtimes b\ltimes$ | |
| $bb$ | $bb$ | $\rtimes bb, bb\ltimes$ | $\rtimes bb\ltimes$ |
| $aaa$ | $aa$ | $\rtimes aa, aaa, aa\ltimes$ | $\rtimes aaa, aaa\ltimes$ |
| $bab$ | $ba, ab$ | $\rtimes ba, bab, ab\ltimes$ | $\rtimes bab, bab\ltimes$ |
| $abba$ | — | $\rtimes ab, abb, bba, ba\ltimes$ | $\rtimes abb, abba, bba\ltimes$ |
| $aabaabaa$ | — | $aab, aba, baa$ | $\rtimes aab, aaba,$ $abaa, baa\ltimes$ |

a. $\forall v_1 v_2 \in \mathtt{fac}_{k-1}(I), v_1 \sigma v_2 \in \mathtt{fac}_k(I)$

b. $\forall v_1 \sigma v_2 \in \mathtt{fac}_{k+1}(I), v_1 v_2 \in \mathtt{fac}_k(I)$

# The algorithm (example)

**Target:** $G_* = \langle T_* = \{b\}, R_* = \{bbb\}\rangle, \Sigma = \{a, b\}, k = 3$
**Sample:** $\{\lambda, a, b, bb, aaa, bab, abba, aabaabaa\}$

| String | 2-factors | 3-factors | 4-factors |
|---|---|---|---|
| $\lambda$ | $\rtimes\ \ltimes$ | | |
| $a$ | $\rtimes a, a\ltimes$ | $\rtimes a \ltimes$ | |
| $b$ | $\rtimes b, b\ltimes$ | $\rtimes b\ltimes$ | |
| $bb$ | $bb$ | $\rtimes bb, bb\ltimes$ | $\rtimes bb\ltimes$ |
| $aaa$ | $aa$ | $\rtimes aa, aaa, aa\ltimes$ | $\rtimes aaa, aaa\ltimes$ |
| $bab$ | $ba, ab$ | $\rtimes ba, bab, ab\ltimes$ | $\rtimes bab, bab\ltimes$ |
| $abba$ | — | $\rtimes ab, abb, bba, ba\ltimes$ | $\rtimes abb, abba, bba\ltimes$ |
| $aabaabaa$ | — | $aab, aba, baa$ | $\rtimes aab, aaba,$ $abaa, baa\ltimes$ |

a. $\forall v_1 v_2 \in \mathtt{fac}_{k-1}(I), v_1 \sigma v_2 \in \mathtt{fac}_k(I)$

b. $\forall v_1 \sigma v_2 \in \mathtt{fac}_{k+1}(I), v_1 v_2 \in \mathtt{fac}_k(I)$

# The algorithm (example)

**Target:** $G_* = \langle T_* = \{b\}, R_* = \{bbb\} \rangle, \Sigma = \{a, b\}, k = 3$
**Sample:** $\{\lambda, a, b, bb, aaa, bab, abba, aabaabaa\}$

| String | 2-factors | 3-factors | 4-factors |
|---|---|---|---|
| $\lambda$ | $\rtimes\ \ltimes$ | | |
| $a$ | $\rtimes a, a\ltimes$ | $\rtimes a \ltimes$ | |
| $b$ | $\rtimes b, b\ltimes$ | $\rtimes b \ltimes$ | |
| $bb$ | $bb$ | $\rtimes bb, bb\ltimes$ | $\rtimes bb \ltimes$ |
| $aaa$ | $aa$ | $\rtimes aa, aaa, aa\ltimes$ | $\rtimes aaa, aaa\ltimes$ |
| $bab$ | $ba, ab$ | $\rtimes ba, bab, ab\ltimes$ | $\rtimes bab, bab\ltimes$ |
| $abba$ | — | $\rtimes ab, abb, bba, ba\ltimes$ | $\rtimes abb, abba, bba\ltimes$ |
| $aabaabaa$ | — | $aab, aba, baa$ | $\rtimes aab, aaba, abaa, baa\ltimes$ |

a. $\forall v_1 v_2 \in \mathtt{fac}_{k-1}(I), v_1 \sigma v_2 \in \mathtt{fac}_k(I)$

b. $\forall v_1 \sigma v_2 \in \mathtt{fac}_{k+1}(I), v_1 v_2 \in \mathtt{fac}_k(I)$

# The algorithm (example)

**Target:** $G_* = \langle T_* = \{b\}, R_* = \{bbb\}\rangle, \Sigma = \{a, b\}, k = 3$
**Sample:** $\{\lambda, a, b, bb, aaa, bab, abba, aabaabaa\}$

| String | 2-factors | 3-factors | 4-factors |
|---|---|---|---|
| $\lambda$ | $\rtimes\ \ltimes$ | | |
| $a$ | $\rtimes a, a\ltimes$ | $\rtimes a \ltimes$ | |
| $b$ | $\rtimes b, b\ltimes$ | $\rtimes b \ltimes$ | |
| $bb$ | $bb$ | $\rtimes bb, bb\ltimes$ | $\rtimes bb\ltimes$ |
| $aaa$ | $aa$ | $\rtimes aa, aaa, aa\ltimes$ | $\rtimes aaa, aaa\ltimes$ |
| $bab$ | $ba, ab$ | $\rtimes ba, bab, ab\ltimes$ | $\rtimes bab, bab\ltimes$ |
| $abba$ | — | $\rtimes ab, abb, bba, ba\ltimes$ | $\rtimes abb, abba, bba\ltimes$ |
| $aabaabaa$ | — | $aab, aba, baa$ | $\rtimes aab, aaba,$ $abaa, baa\ltimes$ |

a. $\forall v_1 v_2 \in \texttt{fac}_{k-1}(I), v_1 \sigma v_2 \in \texttt{fac}_k(I)$

b. $\forall v_1 \sigma v_2 \in \texttt{fac}_{k+1}(I), v_1 v_2 \in \texttt{fac}_k(I)$

# The algorithm (example)

**Target:** $G_* = \langle T_* = \{b\}, R_* = \{bbb\} \rangle, \Sigma = \{a, b\}, k = 3$
**Sample:** $\{\lambda, a, b, bb, aaa, bab, abba, aabaabaa\}$

| String | 2-factors | 3-factors | 4-factors |
|---|---|---|---|
| $\lambda$ | $\rtimes\ \ltimes$ | | |
| $a$ | $\rtimes a, a\ltimes$ | $\rtimes a \ltimes$ | |
| $b$ | $\rtimes b, b\ltimes$ | $\rtimes b \ltimes$ | |
| $bb$ | $bb$ | $\rtimes bb, bb\ltimes$ | $\rtimes bb\ltimes$ |
| $aaa$ | $aa$ | $\rtimes aa, aaa, aa\ltimes$ | $\rtimes aaa, aaa\ltimes$ |
| $bab$ | $ba, ab$ | $\rtimes ba, bab, ab\ltimes$ | $\rtimes bab, bab\ltimes$ |
| $abba$ | — | $\rtimes ab, abb, bba, ba\ltimes$ | $\rtimes abb, abba, bba\ltimes$ |
| $aabaabaa$ | — | $aab, aba, baa$ | $\rtimes aab, aaba,$ $abaa, baa\ltimes$ |

a. $\forall v_1 v_2 \in \mathtt{fac}_{k-1}(I), v_1 \sigma v_2 \in \mathtt{fac}_k(I)$

b. $\forall v_1 \sigma v_2 \in \mathtt{fac}_{k+1}(I), v_1 v_2 \in \mathtt{fac}_k(I)$

# The algorithm (example)

**Target:** $G_* = \langle T_* = \{b\}, R_* = \{bbb\}\rangle, \Sigma = \{a, b\}, k = 3$
**Sample:** $\{\lambda, a, b, bb, aaa, bab, abba, aabaabaa\}$

| String | 2-factors | 3-factors | 4-factors |
|---|---|---|---|
| $\lambda$ | $\rtimes\ltimes$ | | |
| $a$ | $\rtimes a, a\ltimes$ | $\rtimes a\ltimes$ | |
| $b$ | $\rtimes b, b\ltimes$ | $\rtimes b\ltimes$ | |
| $bb$ | $bb$ | $\rtimes bb, bb\ltimes$ | $\rtimes bb\ltimes$ |
| $aaa$ | $aa$ | $\rtimes aa, aaa, aa\ltimes$ | $\rtimes aaa, aaa\ltimes$ |
| $bab$ | $ba, ab$ | $\rtimes ba, bab, ab\ltimes$ | $\rtimes bab, bab\ltimes$ |
| $abba$ | — | $\rtimes ab, abb, bba, ba\ltimes$ | $\rtimes abb, abba, bba\ltimes$ |
| $aabaabaa$ | — | $aab, aba, baa$ | $\rtimes aab, aaba, abaa, baa\ltimes$ |

a. $\forall v_1 v_2 \in \texttt{fac}_{k-1}(I), v_1 \sigma v_2 \in \texttt{fac}_k(I)$

b. $\forall v_1 \sigma v_2 \in \texttt{fac}_{k+1}(I), v_1 v_2 \in \texttt{fac}_k(I)$

# The algorithm (example)

**Target:** $G_* = \langle T_* = \{b\}, R_* = \{bbb\}\rangle, \Sigma = \{a, b\}, k = 3$
**Sample:** $\{\lambda, a, b, bb, aaa, bab, abba, aabaabaa\}$

| String | 2-factors | 3-factors | 4-factors |
|---|---|---|---|
| $\lambda$ | $\rtimes\ltimes$ | | |
| $a$ | $\rtimes a, a\ltimes$ | $\rtimes a\ltimes$ | |
| $b$ | $\rtimes b, b\ltimes$ | $\rtimes b\ltimes$ | |
| $bb$ | $bb$ | $\rtimes bb, bb\ltimes$ | $\rtimes bb\ltimes$ |
| $aaa$ | $aa$ | $\rtimes aa, aaa, aa\ltimes$ | $\rtimes aaa, aaa\ltimes$ |
| $bab$ | $ba, ab$ | $\rtimes ba, bab, ab\ltimes$ | $\rtimes bab, bab\ltimes$ |
| $abba$ | — | $\rtimes ab, abb, bba, ba\ltimes$ | $\rtimes abb, abba, bba\ltimes$ |
| $aabaabaa$ | — | $aab, aba, baa$ | $\rtimes aab, aaba, abaa, baa\ltimes$ |

a. $\forall v_1 v_2 \in \texttt{fac}_{k-1}(I), v_1 \sigma v_2 \in \texttt{fac}_k(I)$

b. $\forall v_1 \sigma v_2 \in \texttt{fac}_{k+1}(I), v_1 v_2 \in \texttt{fac}_k(I)$

# The algorithm (example)

**Target:** $G_* = \langle T_* = \{b\}, R_* = \{bbb\}\rangle, \Sigma = \{a, b\}, k = 3$
**Sample:** $\{\lambda, a, b, bb, aaa, bab, abba, aabaabaa\}$

| String | 2-factors | 3-factors | 4-factors |
|---|---|---|---|
| $\lambda$ | $\rtimes\ltimes$ | | |
| $a$ | $\rtimes a, a\ltimes$ | $\rtimes a\ltimes$ | |
| $b$ | $\rtimes b, b\ltimes$ | $\rtimes b\ltimes$ | |
| $bb$ | $bb$ | $\rtimes bb, bb\ltimes$ | $\rtimes bb\ltimes$ |
| $aaa$ | $aa$ | $\rtimes aa, aaa, aa\ltimes$ | $\rtimes aaa, aaa\ltimes$ |
| $bab$ | $ba, ab$ | $\rtimes ba, bab, ab\ltimes$ | $\rtimes bab, bab\ltimes$ |
| $abba$ | — | $\rtimes ab, abb, bba, ba\ltimes$ | $\rtimes abb, abba, bba\ltimes$ |
| $aabaabaa$ | — | $aab, aba, baa$ | $\rtimes aab, aaba, abaa, baa\ltimes$ |

a. $\forall v_1 v_2 \in \mathtt{fac}_{k-1}(I), v_1 \sigma v_2 \in \mathtt{fac}_k(I)$

b. $\forall v_1 \sigma v_2 \in \mathtt{fac}_{k+1}(I), v_1 v_2 \in \mathtt{fac}_k(I)$

# The algorithm (example)

**Target:** $G_* = \langle T_* = \{b\}, R_* = \{bbb\}\rangle, \Sigma = \{a, b\}, k = 3$
**Sample:** $\{\lambda, a, b, bb, aaa, bab, abba, aabaabaa\}$

| String | 2-factors | 3-factors | 4-factors |
|---|---|---|---|
| $\lambda$ | $\rtimes\ltimes$ | | |
| $a$ | $\rtimes a, a\ltimes$ | $\rtimes a\ltimes$ | |
| $b$ | $\rtimes b, b\ltimes$ | $\rtimes b\ltimes$ | |
| $bb$ | $bb$ | $\rtimes bb, bb\ltimes$ | $\rtimes bb\ltimes$ |
| $aaa$ | $aa$ | $\rtimes aa, aaa, aa\ltimes$ | $\rtimes aaa, aaa\ltimes$ |
| $bab$ | $ba, ab$ | $\rtimes ba, bab, ab\ltimes$ | $\rtimes bab, bab\ltimes$ |
| $abba$ | — | $\rtimes ab, abb, bba, ba\ltimes$ | $\rtimes abb, abba, bba\ltimes$ |
| $aabaabaa$ | — | $aab, aba, baa$ | $\rtimes aab, aaba, abaa, baa\ltimes$ |

- $T = \{b\}, R = \texttt{fac}_3(T^*) - \texttt{fac}_3(I) = \{bbb\}$

# The algorithm (correctness)

**The non-tier member lemma:** Iff $\sigma \notin T$:

a. $\forall v_1 v_2 \in \texttt{fac}_{k-1}(L(G)), v_1 \sigma v_2 \in \texttt{fac}_k(L(G))$

b. $\forall v_1 \sigma v_2 \in \texttt{fac}_{k+1}(L(G)), v_1 v_2 \in \texttt{fac}_k(L(G))$

**The characteristic sample** is a set $C$ such that

- For every $\sigma \notin T$,
  - $\forall v_1 v_2 \in \texttt{fac}_{k-1}(L), \exists v_1 \sigma v_2 \in \texttt{fac}_k(C)$.
  - $\forall v_1 \sigma v_2 \in \texttt{fac}_{k+1}(L), \exists v_1 v_2 \in \texttt{fac}_k(C)$.

# The algorithm (correctness)

**The non-tier member lemma:** Iff $\sigma \notin T$:

a. $\forall v_1 v_2 \in \texttt{fac}_{k-1}(L(G)), v_1 \sigma v_2 \in \texttt{fac}_k(L(G))$

b. $\forall v_1 \sigma v_2 \in \texttt{fac}_{k+1}(L(G)), v_1 v_2 \in \texttt{fac}_k(L(G))$

**The characteristic sample** is a set $C$ such that

- For every $\rho \in T$ that appears in $R$, some $v_1 v_2 \in \texttt{fac}_{k-1}(C)$ such that $v_1 \rho v_2 \in R$
- For all other $\tau \in T$, some $v_1 \tau v_2 \in \texttt{fac}_{k+1}(C)$ such that $v_1 v_2 \in R$

# The algorithm (correctness)

**The characteristic sample** is a set $C$ such that

- For every $w \in \texttt{fac}_k(T^*) - R$, $w \in \texttt{fac}_k(C)$

# The algorithm (data complexity)

- The minimum size of the characteristic sample is bounded by $\mathcal{O}(|\Sigma|^k)$, which is constant

# The algorithm (time complexity)

- For input $I$ and $n = ||I||$, the $k$TSLIA runs in $\mathcal{O}(n^2)$ time
- Complexity of $\texttt{fac}_{k(\pm 1)}(I)$ is $\mathcal{O}(n)$
- Two main steps:

  a. $$\underbrace{\forall v_1 v_2 \in \texttt{fac}_{k-1}(I)}_{\mathcal{O}(n)}, \underbrace{v_1 \sigma v_2 \in \texttt{fac}_k(I)}_{\mathcal{O}(n)} \quad \mathcal{O}(n) \cdot \mathcal{O}(n) = \mathcal{O}(n^2)$$

  b. $$\underbrace{\forall v_1 \sigma v_2 \in \texttt{fac}_{k+1}(I)}_{\mathcal{O}(n)}, \underbrace{v_1 v_2 \in \texttt{fac}_k(I)}_{\mathcal{O}(n)} \quad \mathcal{O}(n) \cdot \mathcal{O}(n) = \mathcal{O}(n^2)$$

- One more scan through $\texttt{fac}_k(I)$ (to find $R$) $= \mathcal{O}(n)$

# Discussion and conclusion

- Given $k$, the $k$TSLIA exactly identifies any $\mathrm{TSL}_k$ language in quadratic time with a characteristic sample bounded in size by a constant w.r.t. that language's grammar
- The $k$TSLIA built on specific properties of elements of $T$ and $T - \Sigma$

# Discussion and conclusion

- Given $k$, the $k$TSLIA exactly identifies any TSL$_k$ language in quadratic time with a characteristic sample bounded in size by a constant w.r.t. that language's grammar
- The $k$TSLIA built on specific properties of elements of $T$ and $T - \Sigma$
- This result motivated by natural language phonotactics
  - Is the $I_C$ present in natural language data?
  - How can a stochastic learner build on this $k$TSLIA?
  - How can we extend these ideas to phonological *functions* (e.g. [JAKC15, CJH15])?

# Thank you!

We would also like to thank Gunnar Hansson, Jane Chandlee, Jeffrey Heinz, and three anonymous reviewers for their thoughts and insights.

# References I

[CJH15] Jane Chandlee, Adam Jardine, and Jeffrey Heinz, *Learning repairs for marked structures*, Proceedings of the 2015 Annual Meeting on Phonology, LSA, 2015.

[Coh92] Abigail Cohn, *The consequences of dissimilation in Sundanese*, Phonology **9** (1992), 199–220.

[CS82] George N. Clements and Engin Sezer, *Vowel and consonant disharmony in Turkish*, The Structure of Phonological Representations (Part II) (Harry van der Hulst and Norval Smith, eds.), Foris, Dordrecht, 1982.

[Cse10] András Cser, *The -alis/aris- allomorphy revisited*, Variation and change in morphology: selected papers from the 13th international morphology meeting (D. Kastovsky, F. Rainer, W. U. Dressler, and H. C. Luschützky, eds.), Philadelphia: John Benjamins, 2010, pp. 33–51.

[dlH97] Colin de la Higuera, *Characteristic sets for polynomial grammatical inference*, Machine Learning **27** (1997), no. 2, 125–138.

# References II


[Gol67] Mark E. Gold, *Language identification in the limit*, Information and Control **10** (1967), 447–474.

[Hei10] Jeffrey Heinz, *Learning long-distance phonotactics*, LI **41** (2010), 623–661.

[Hei11] ______, *Computational phonology part I: Foundations*, Language and Linguistics Compass **5** (2011), no. 4, 140–152.

[HRT11] Jeffrey Heinz, Chetan Rawal, and Herbert G. Tanner, *Tier-based strictly local constraints for phonology*, Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (Portland, Oregon, USA), Association for Computational Linguistics, June 2011, pp. 58–64.

[JAKC15] Adam Jardine, Angeliki Athanasopoulou, Kristian, and Peter Cole, *Banyaduq prestopped nasals: Synchrony and diachrony*, Oceanic Linguistics **54** (2015), 548–578.

[JH16] Adam Jardine and Jeffrey Heinz, *Learning tier-based strictly 2-local languages*, Transactions of the Association for Computational Linguistics **4** (2016), 87–98.

# References III


[MH16] Kevin McMullin and Gunnar Ólafur Hansson, *Long-distance phonotactics as Tier-Based Strictly 2-Local languages*, Proceedings of the Annual Meeting on Phonology 2015 (Adam Albright and Michelle A. Fullwood, eds.), 2016.

[MP71] Robert McNaughton and Seymour Papert, *Counter-free automata*, MIT Press, 1971.

[Nev10] Andrew Nevins, *Locality in vowel harmony*, Linguistic Inquiry Monographs, no. 55, MIT Press, 2010.

[Odd94] David Odden, *Adjacency parameters in phonology*, Language **70** (1994), no. 2, 289–330.

[RHF+13] James Rogers, Jeffrey Heinz, Margaret Fero, Jeremy Hurst, Dakotah Lambert, and Sean Wibel, *Cognitive and sub-regular complexity*, Formal Grammar, Lecture Notes in Computer Science, vol. 8036, Springer, 2013, pp. 90–108.

[Rin75] Catherine Ringen, *Vowel harmony: Theoretical implications*, Ph.D. thesis, Indiana University, 1975.

[RW04] Sharon Rose and Rachel Walker, *A typology of consonant agreement as correspondence*, Language **80** (2004), 475–531.