Learning within- and between-word variation in probabilistic OT grammars

Aleksei Nazarov
University of Huddersfield

1 Introduction

This paper proposes a novel method of inferring diacritics for representing between-word variation (exceptionality) in Optimality Theoretic (OT) grammars (e.g., Pater 2000, 2010) that makes it possible to infer such diacritics in the face of within-word variation. Existing methods of inferring diacritics in OT (Pater 2010, Becker 2009, Coetzee 2009) are based in categorical grammar learning (Tesar 1995), which makes them unable to handle within-word variation. Existing methods of inferring probabilistic OT grammars (e.g., Boersma 1998) handle within-word variation well, but have no provision to distinguish exceptional from non-exceptional words, and are incompatible with the main idea in Pater (2010). I show that this latter idea can be made compatible with probabilistic grammars, so that both within- and between-word variation can be learned. Before launching into the details of the learning algorithm, I will define and exemplify both types of variation – in isolation as well as in combination.

1.1 *Types of variation* Phonological processes allows variation of different kinds, as summarized by, e.g., Coetzee and Pater (2011). One type may be called within-word variation, which may also be called optionality or variability: this is variation in which the same underlying sound, /X/, may be pronounced differently in the same context and in the same word, as in (1a). This is exemplified by English optional phrasal nasal place assimilation (see (2a)) where any word ending underlyingly in /n/ maybe be realized [n] or [m] before a labial stop or nasal.

Another type may be called between-word variation, which may also be called exceptionality: variation where the same underlying sound, /X/, is pronounced differently in the same context but in different words, as in (1b). The example given in (2b) is Dutch primary stress, which is always penultimate in some words, but is always antepenultimate in other words.

Most importantly, both types of variation may co-occur within the same process in the same language, as schematized in (1c). Such a situation is observed in Modern Hebrew, (2c), which has a spirantization process that turns plosives /p,b,k/ into fricatives after a vowel. This process standardly undergoes within-word variation, as in (2c.i): the underlying plosives /p,b,k/ are free to surface faithfully as plosives, or as their corresponding fricatives, [f, v, χ]. However, there is also between-word variation: some words deviate from this pattern of optionality by either disallowing spirantization, as in (2c.ii), or requiring spirantization, as in (2c.iii).

(1) *Summary of the three types of variation*

a. Within-word	*b. Between-word*	*c. Within- and between-word*
$X \rightarrow Y \sim Z \quad / \alpha\_\beta$	$X_{word1} \rightarrow Y \quad / \alpha\_\beta$	$X_{word1} \rightarrow Y \sim Z \quad / \alpha\_\beta$
	$X_{word2} \rightarrow Z \quad / \alpha\_\beta$	$X_{word2} \rightarrow Z \quad / \alpha\_\beta$

* Many thanks to Adam Albright, Ricardo Bermúdez-Otero, Paul Boersma, Silke Hamann, Bruce Hayes, Gaja Jarosz, Tal Linzen, Wendell Kimper, Giorgio Magri, John McCarthy, Marc van Oostendorp, Joe Pater, Juliet Stanton, Robert Staubs, Patrycja Strycharczuk, Kristine Yu, and audiences at UMass Amherst, the University of Amsterdam, MIT, the University of Connecticut, the University of Huddersfield, and the University of Manchester, as well as at the 23rd Manchester Phonology Meeting, the LSA 2016 Annual Meeting, the Annual Meeting in Phonology 2017, and the Society for Computation in Linguistics 2018 Meeting for very helpful and stimulating discussion of various instantiations of this work.

Proceedings of *AMP 2017*

(2) *Examples of the types of variation*

a. English nasal assimilation (Coetzee and Pater 2011)	*b. Dutch primary stress* (Kager 1989)	*c. Hebrew spirantization* (Temkin-Martínez 2010)
/n/ → [n ~ m] / _[+lab,–cont]	$CVCV_{word1}$ → CVCV́ / _CV# $CVCV_{word2}$ → CV́CV / _CV#	i. /p,b,k/ → [p ~ f, b ~ v, k ~ χ] / V_ ii. /p,b,k/ → [p,b,k] / V_ iii. /p,b,k/ → [f,v,χ] / V_
/gɹin bɑks/ → [gɹi**n** bɑks] ~ [gɹi**m** bɑks] /ɪn bɛd/ → [ɪ**n** bɛd] ~ [ɪ**m** bɛd]	/kasino/ → [**ka.**ˈ**si**.no] /kimono/ → [ˈ**ki.mo**.no]	/me**k**ase/ → [me**k**ase~me**χ**ase] /da**k**ar/ → [da**k**ar] ***χ** /ma**k**ar/ → [ma**χ**ar] ***k**

1.2 *Implications for learning* The possibility of co-existence of within-word and between-word variation (as, for instance, in Modern Hebrew – see (2c) above) has implications for human and machine learners of phonology. Children do not know in advance which words in a language's lexicon are phonologically exceptional, so that they must be able to somehow notice that some groups of words behave differently from other groups of words and then infer diacritics for each group. At the same time, an adult grammar of a language like Modern Hebrew must be able to account for the spirantization behavior of both exceptional and non-exceptional words, including the (within-word) variability of applying spirantization in non-exceptional words.

As I will review in section 2, computational learners have been proposed for within-word variation separately (including Boersma 1998 and Jarosz 2015 for Optimality Theory), and between-word variation separately (Pater 2010, Becker 2009, Coetzee 2009), but not for dealing with both kinds of variation within the same dataset. However, a learner that can deal with both between- and within-word variation in the same dataset hasn't been proposed before – with the exception of Moore-Cantwell (2017), a learning proposal that can represent between-word in the face of within-word variation when between-word variation can be marked in the underlying form without the need of diacritics. There are no previous proposals for the more general case, when between-word variation cannot necessarily be marked in the underlying form without diacritics.

1.3 *Proposal* As will be shown in section 2.2, the criterion proposed by Pater (2010), Becker (2009), and Coetzee (2009) for initiating the induction of an exceptionality category is not compatible with a probabilistic framework (such as Boersma 1998 or Jarosz 2015). My proposal, then, is to slightly modify this criterion to make it compatible with a probabilistic framework – while the original criterion is called Inconsistency (Tesar 1995), I propose to make this criterion probabilistic and call it Soft Inconsistency (see section 3). I will then go on and demonstrate the effectiveness of this criterion on a simplified Modern Hebrew spirantization dataset. The rest of this paper is structured as follows: section 2 will detail existing work on learning within- and between-word variation; section 3 will spell out my own proposal of a Soft Inconsistency criterion, after which section 4 will explain how this proposal was implemented in the simulations reported on in this paper; section 5 will finally show results from simulations with this learner run on somewhat simplified Hebrew spirantization, after which section 6 will offer some concluding remarks.

2 Previous work: Probability vs. inconsistency

2.1 *Probabilistic models: degrees of match* Classic OT (Prince and Smolensky 1993/2004) requires that one input be mapped to the same output at all times, so that within-word variation is out of the question (although see Anttila 1997 for an approach within classic OT that uses ties to account for a limited range of within-word variation). However, there are various probabilistic versions of OT that do allow within-word variation. These models have in common that they are non-deterministic: they allow one input to yield different outputs depending on a stochastic component.

Probably the simplest models of within-word variation in OT are the Partially Unordered Grammar model (Anttila 2002) and its closely related approach of Floating Constraints (Nagy and Reynolds 1997). In

these approaches, the mutual ranking of certain constraints a language is underspecified, and every time the grammar is consulted the underspecified rankings are filled in *ad libitum*. For instance, if the grammar has constraints {A, B, C}, then the grammar may contain rankings A >> C and B >> C, but no specification for the mutual ranking of constraints {A, B}. For this reason, the grammar may use the ranking A >> B >> C at one time, and the ranking B >> A >> C at another time.

More statistically advanced models are Stochastic OT (Boersma 1998) and Pairwise Ranking Grammar (PRG; Jarosz 2015). In both of these approaches, there is a statistical model that samples a constraint ranking every time the grammar is consulted. For instance, with constraints {A, B, C}, the model may sample the ranking A >> B >> C at one time, and the ranking B >> A >> C at another time.

The difference between Stochastic OT and PRG lies in the specific statistical model that generates these rankings. In Stochastic OT, constraint ranking is derived from numeric weights for every constraint; these numeric weights are perturbed by the addition of Gaussian noise. For instance, let us suppose that the model specifies weights $w(A) = 2$ and $w(B) = 1$, which standardly yields the ranking A >> B, since 2 > 1. Because of noise, these two constraints may end up with weights $w(A) = 1.49$ and $w(B) = 1.51$, which yields the opposite ranking, B >> A, since 1.51 > 1.49. In this manner, the proximity of two constraint weights and the extent of added noise may lead to differences in ranking of the same constraints.

In PRG, however, ordinal constraint ranking is derived from probabilities over pairwise rankings. For instance, for constraints {A, B, C}, there are three unique pairs of constraints: {A, B}, {B, C}, and {A, C}. PRG specifies a binomial probability for both possible rankings of each pair. For instance, for the pair {A, B}, it may specify the probability P(A>>B) = 0.6. This means that there is a 60% chance of sampling the ranking A >> B, and a 40% chance of sampling the ranking B >> A. The sampling procedure for entire constraint rankings is somewhat more complex, since independent sampling of pairwise rankings may yield logically impossible results. For instance, P(A>>B) = P(B>>C) = P(A>>C) = 0.5, then one may draw from each of these pairwise distributions the rankings A >> B, B >> C, and C >> A, which cannot be assembled into a congruent ranking. Rather, the sampling procedure proposed by Jarosz (2015) goes through all pairwise rankings in a random order, and each time computes the probability of that ranking given all other rankings that have been sampled at previous steps. An example of this can be found in the Appendix.

Two other popular models of within-word variation closely related to OT are Noisy Harmonic Grammar (Noisy HG; Coetzee and Pater 2011) and Maximum Entropy models (MaxEnt models; Goldwater and Johnson 2003). Noisy HG operates precisely as Stochastic OT, except that the numeric weights for every constraint are not converted into a categorical ranking, leading to the possibility of cumulative constraint interaction (Potts et al. 2009). In MaxEnt, constraints also have numeric weights, but these are not perturbed by noise, but a statistical distribution over outputs is computed by entering the constraints into a multinomial logistic regression model (Manning and Schutze 1999).

Within the ranked-constraint probabilistic frameworks mentioned here, Stochastic OT and PRG, learning proceeds along the following lines (for more details, see Boersma 1998 for the Gradual Learning Algorithm developed for Stochastic OT, and see Jarosz 2015 as well as the Appendix for the Expectation Driven Learning algorithm developed for PRG). Whenever the learner sees a data token that is only consistent with the ranking A >> B, it will increase the probability of sampling A >> B – either by increasing the weight of A relative to the weight of B (for Stochastic OT), or by increasing P(A>>B) directly (as in the case of PRG). Whenever the learner see a data token that is only consistent with the ranking B >> A, conversely, the probability of sampling A >> B will decrease.

Consequently, if the data contain tokens only consistent with A >> B as well as tokens only consistent with B >> A, then a learner that starts with P(A>>B) = 0.5 will respond to these data by increasing P(A>>B) in some cases and decreasing P(A>>B) in other cases, yielding a value of P(A>>B) between 0 and 1. Such a value of P(A>>B) will lead to within-word variation, since any input will be produced with the ranking A >> B in some cases, and with the ranking B >> A in other cases. Thus, contradictory ranking requirements (A >> B for some tokens, B >> A elsewhere) provide evidence for within-word variation.

No previous mechanism has been proposed to factor out between-word variation in a probabilistic learner – the only exception being Moore-Cantwell (2017), who proposes that phonological features in Underlying Representations (URs) may have different "strength", corresponding roughly to activation in memory (see also Smolensky et al. 2014). In a MaxEnt framework, words with "strong" features of certain kinds may exert a stronger Faithfulness effect than other words, allowing an exceptional phonological form to words with "strong" features. This allows certain patterns of exceptionality to the represented and

learned, and since MaxEnt models are able to represent within-word variation as well, this model can combine within-word variation with at least certain kinds of between-word variation.

However, not all patterns of exceptionality can be represented with substantive features in URs. Mullin (2012) and Osadcha (2014), among others, present patterns in which the contrast between various kinds of exceptional patterns and the default pattern cannot be reduced to a difference in substantive features between their URs, even when underspecification (see, e.g., Inkelas et al.'s 1997 account of exceptionality in Turkish final devoicing) is considered – despite Kim and Pulleyblank's (2009) argument that all exceptionality must be represented with substantive features in the UR. The proposal in section 3 is aimed at exactly such cases in which between-word variation must be represented by an arbitrary diacritic rather than a substantive phonological feature. As I will argue in section 5, Hebrew spirantization is one of these cases, which is one of the reasons to test the current learner on this case study.

2.2 *Categorical models: inconsistency* As mentioned at the beginning of section 2.1, Classic OT (Prince and Smolensky 1993/2004) only allows one output per input, as it is a categorical, non-probabilistic model. Existing work on inferring diacritics to represent between-word variation (Pater 2010, Becker 2009, Coetzee 2009; see section 4.1 for the types of diacritics they infer) is set in this framework, and, as I will show here, its principles are rooted in the categorical nature of Classic OT.

Pater (2010), Becker (2009), and Coetzee (2009) base their learner for inferring between-word variation diacritics in Recursive Constraint Demotion (RCD; Tesar 1995). RCD is a technique that infers from a data set a (partial) ranking of constraints that is consistent with the data. It builds this ranking up in steps, and at every step it asks for constraints A and B whether there are data points (in the form of pairs of a winning candidate and a corresponding losing candidate) that are only consistent with ranking A >> B. If this is so, and there are no data points that require B >> A, then ranking A >> B is added to the grammar. Thus, it is a technique based on the idea of logical consistency.

RCD is guaranteed to find a (partial) ranking that generates the data set as long as there are no data points such that one data point requires A >> B and the other data point requires B >> A. If the latter scenario does occur, then *inconstistency* is declared, and RCD is stopped without finding a complete solution.

Pater (2010) proposes that, instead, this state of *inconsistency* be the trigger for inferring diacritics to represent between-word variation. Once a diacritic and a corresponding constraint (see section 4.1) are inferred, inconsistency is removed from the learning problem, and RCD can resume its construction of a constraint ranking. Becker (2009) and Coetzee (2009) show particular implementations of this idea, with specific algorithms that turn the idea that there are exceptional words into the addition of specific diacritics and concomitant constraints.

Thus, the probabilistic OT learners mentioned in section 2.1 and the categorical learners described in this section have an opposite response to opposite ranking requirements, as shown in the summary of the difference between the learners in table (3) below. The probabilistic learners respond to opposite ranking requirements by assuming probabilistic ranking without assuming any inconsistency, while the categorical learners respond by declaring inconsistency and assuming that there is between-word variation. This makes the two approaches incompatible in their current form, and makes neither model able to learn both within- and between-word variation. In section 3, I will show my proposal of how to unify the two approaches.

(3) *Comparison between existing probabilistic and categorical OT learners*

	Probabilistic OT learners (Boersma 1998, Jarosz 2015)	*Categorical OT learners* (Tesar 1995, Becker 2009, Coetzee 2009)
Can learn within-word variation	Yes	No
Can learn between-word variation	No	Yes
Deterministic model?	Non-deterministic. Rankings sampled from statistical distribution.	Deterministic. Single ranking used at all times.
Whenever a word requires A >> B:	P(A >> B) increased	P(A >> B) set to 1
Whenever word$_1$ requires A >> B and word$_2$ requires B >> A:	0 < P(A >> B) < 1 (leading to within-word variation)	Inconsistency detected (leading to diacritics)

3 Proposal: Soft (or probabilistic) inconsistency detection

I propose here to combine insights from both types of learners described in section 2 to yield a model that can learn both within-word and between-word variation. My proposal is to infer the existence of between-word variation from *soft inconsistency*, with the gist that opposite ranking *tendencies* between words – rather than opposite ranking *requirements* between words – are the mechanism behind detecting between-word variation. While regular inconsistency detection in categorical OT learning (Tesar 1995) may be used to trigger the assumption of between-word variation and the induction of diacritics for exceptional words (Pater 2010, Becker 2009, Coetzee 2009), I will show that soft inconsistency detection can trigger the same effect in probabilistic grammars. Details of a particular implementation of the proposal will be given in section 4, while the results of tests of this implementation on the Hebrew example sketched in (2c) in section 1 will be reported in section 5.

The fundamental basic requirement for the soft inconsistency proposal is to distinguish between word-tokens (specific pronunciations/outputs for a certain word) and word-types (the underlying words/inputs that word-tokens are realizations of). This is necessary to be able to discriminate within-word variation (where the difference is among tokens of the same word-type) from between-word variation (where the difference is among word-types). In practical terms, this means probabilities of rankings must be computed per word (for determining between-word variation) as well for the entire dataset (for determining within-word variation, since this has to be consistent across the lexicon).

(4) *Adjusting probabilistic learners to distinguish between word-tokens and word-types*

a. *Regular probabilistic grammars*

Estimate P(A >> B | data set),
P(B >> C | data set), ...

b. *Type/token-sensitive probabilistic grammars*

Estimate $P(A >> B \mid word_1)$, $P(A >> B \mid word_2)$, ..., $P(A >> B \mid word_n)$, P(A >> B | data set),
$P(B >> C \mid word_1)$, ..., $P(B >> C \mid word_n)$, P(B >> C | data set), ...

Building on this type/token distinction, (classic) inconsistency can be defined in a probabilistic setting: it is the situation in which, for the same constraint pair {A, B}, the probability of A >> B equals 1 for one word, and the probability of that same ranking equals 0 for another word. This is schematically represented in (5a) below. Compare this to the definition in section 2.2: "one data point requires A >> B and the other data point requires B >> A".

However, in probabilistic grammars, the pairwise ranking probability of two constraints will rarely be exactly 1 or 0. In fact, in Stochastic OT, this probability is never exactly 1 or 0.[1] Therefore, I propose to weaken the inconsistency requirement in (5a), which represents opposite ranking requirements, to (5a), which represents opposite ranking preferences. I define a ranking preference for ranking A >> B as a 50% or larger probability of A >> B. Based on this definition, I propose that soft inconsistency be declared whenever one word has a ranking preference for A >> B and another word has a ranking preference for B >> A.

(5) *Comparison between regular inconsistency and soft inconsistency*

a. *Inconsistency (see section):*

$P(A >> B) = 1$ for $word_1$ & $P(A >> B) = 0$ for $word_2$

b. *Soft inconsistency (= probabilistic inconsistency):*

$P(A >> B) > 0.5$ for $word_1$ & $P(A >> B) < 0.5$ for $word_2$

In order to properly assess ranking preference, I propose to add a fixed threshold to 50% – in the simulations described in section 5, I used a 10% threshold. Therefore, the formulation of soft inconsistency used in these simulation is that P(A>>B) had to be above 0.6 for $word_1$ and below 0.4 for $word_2$. Thus, if

[1] Gaussian noise is, technically speaking, unbounded, so that constraints {A,B} with weights $w(A) = 100{,}000$ and $w(B) = 0$ and Gaussian noise with a standard deviation of 0.1 still have a very small chance of ending up ranked as B >> A.

within a dataset, there is a word that yields a 65% probability of, for instance, *Stop >> *Fricative, while there is another word that yields only a 35% probability of *Stop >> *Fricative, there learner declares there to be soft inconsistency, meaning that there is a reason to assume there is between-word variation. Thus, whenever the learner detects soft inconsistency, it has evidence for between-word variation in the data – just like regular inconsistency triggers the inference of a diacritic in Pater's (2010), Becker's (2009), and Coetzee's (2009) proposals. See section 4.2 for the precise procedure followed for inducing and maintaining diacritics.

A specific implementation of soft inconsistency detection could be constructed in a range of different probabilistic variants of OT and a range of different learning mechanisms, as long as the following minimal requirements are satisfied:

(6) *Minimal requirements for using soft inconsistency*
1. Access to pairwise ranking probability estimates (necessary for computing ranking preferences)
2. Possibility of computing such probabilities for individual words/inputs (for comparing words)

While these minimal requirements could be implemented in various learners, for instance, the Gradual Learning Algorithm for Stochastic OT (Boersma 1998), they are most straightforward to implement in Expectation Driven Learning (EDL, Jarosz 2015; see extra information in the Appendix). This learner directly estimates P(A >> B) to learn a Pairwise Ranking Grammar, and these estimates can be computed based on an individual word as well as based on the entire lexicon. This satisfies the criteria in (6) without any additional tweaking, which is why it was chosen here. The specific grammar framework used included Indexed Constraint Theory (Kraska-Szlenk 1995, Pater 2000, 2010), which is just one way of representing between-word variation in OT – the major competitor being Cophonology Theory (e.g., Inkelas 1998). The implementation chosen here will be described in detail in section 4.

4 Current implementation

4.1 *Implementation of between-word variation diacritics* The model of between-word variation used here is Indexed Constraint Theory (Kraska-Szlenk 1995, Pater 2000, 2010). This model was chosen partly because it was also used by Pater (2010), Becker (2009), and Coetzee (2009). In Indexed Constraint Theory, one and the same constraint may have several instantiations: one general version, and versions with various diacritics (indices). Indexed versions of the constraint only have violations for inputs that carry the index specified on the constraint. For instance, in tableaux (7b) below, the indexed constraint Ident_i with index *i* has violations only for the input /akpa$_i$/, which also carries index *i* – while for the input /akta/ in tableau (7a), which does not carry index *i*, the constraint Ident_i has no violations.

Such a system can lead essentially to reversals of certain rankings for indexed items. For instance, the grammar in (7ab) specifies that *Stop/V_ (no stops after a vowel) is ranked above Ident, leading to spirantization after a vowel in (7a). However, the indexed constraint Ident_i essentially reverses this ranking for words that carry the index *i*: spirantization is blocked in /akpa$_i$/ because of the ranking Ident_i >> *Stop/V_, as illustrated in (7b) – even though non-indexed Ident is still ranked below *Stop/V_.

(7) *Illustration of Indexed Constraint Theory*

a. *Non-indexed input*

/akta/	Ident_i	*Stop/V_	Ident
[akta]		*!	
☞[axta]			*

b. *Indexed input*

/akpa$_i$/	Ident_i	*Stop/V_	Ident
☞ [akpa]		*	
[axpa]	*!		*

In this paper, I will assume with Pater (2000, 2010) that every constraint has an unindexed variant that are violated for all inputs, contra Becker (2009). For instance, if there is a constraint Ident_i that is violated

only for inputs with index *i*, this entails the existence of a constraint Ident which is violated for any input. This means that between-word variation is interpreted as a distinction between default-obeying words (those that have no index) and exceptional words (those that do have an index). This view of a default pattern vs. (an) exceptional pattern(s) has consequences for how soft inconsistency is assessed, as detailed in the next subsection, but the general approach described here is certainly also compatible with a view where no pattern is seen as inherently more default or standard, and, instead, all available patterns are seen as being on a par (see Becker 2009, as well as work in Cophonology Theory, e.g., Inkelas 1998).

4.2 *Implementation of soft inconsistency and inference of diacritics* As indicated in section 4.1, between-word variation was interpreted in this case as a distinction between inputs without an index (the default-obeying words) and inputs with some index (the exceptional words), following Pater (2000, 2010). This has consequences for the implementation of soft inconsistency. If what is to be detected is not simply the existence of between-word variation, but specifically the presence of exceptions to a default pattern, the detection procedure must involve assessing whether an individual word fits with the tendencies observed across the entire lexicon, instead of simply comparing individual words. This is because of the assumption inherent to the default vs. exceptions view that all words in the lexicon belong to the default unless otherwise specified.

For this reason, soft inconsistency was implemented as a comparison between the ranking tendencies in the entire lexicon and the ranking tendencies of individual words. As mentioned earlier in section 3, a threshold of 10% above or below 50% was used in assessing whether these tendencies were truly in the opposite direction. The resulting precise formulation of soft inconsistency used in the simulations for this paper is as follows:

(8) *Definition of soft inconsistency used in the current simulations*
$Word_n$ is soft-inconsistent with the lexicon on constraint pair {A,B} iff ...
$P(A >> B) > 0.6$ for $word_n$ & $P(A >> B) < 0.4$ across the lexicon

This definition of soft inconsistency was embedded in a batch Expectation Driven Learning framework (Jarosz 2015), which learns pairwise ranking probabilities (see section 2.1) through multiple iterations of Expectation Maximization (Dempster et al. 1977) re-estimation, starting with P(X>>Y) = P(Y>>X) = 0.5 for all constraint pairs {X,Y}. At each iteration, pairwise ranking probabilities were computed given each individual word as well as given the entire lexicon, after which soft inconsistency, as defined in (8), was assessed for each word and each constraint pair.

As in Pater (2010), Becker (2009), Coetzee (2009), the inconsistency criterion led to induction of indexed constraints (see section 4.1). It is not the case, however, that every word that was soft-inconsistent with some constraint pair at some iteration was assigned to an indexed constraint. Rather, at most one indexed constraint was induced per iteration of the learning algorithm – which was done in order to minimize the number of "false alarms": words that appear exceptional at first sight, but fall within the default pattern once the learner has built a more advanced grammar.

At every iteration, only the "most exceptional" constraint pair was selected for indexed constraint induction. For every constraint pair {X,Y} and every input word $word_{inc(XY)}$ soft-inconsistent with it, the absolute difference between P(X>>Y) for $word_{inc(XY)}$ and P(X>>Y) across the lexicon was computed. The "most exceptional" constraint pair was the one with the highest sum of these absolute ranking tendency differences – which either means that this constraint pair has the most exceptional words associated with it, or that the exceptional words have the strongest opposition to the default pattern, or both.

Once this "most exceptional" constraint pair was identified, then a relevant indexed constraint was inferred, if this particular indexed constraint did not exist yet.[2] The constraint that was given an indexed version was always the constraint that was preferred to be higher-ranked in the exceptional words. For instance, if the constraint pair was {Ident, *Stop/V_}, and the soft-inconsistent words preferred Ident >> *Stop/V_, then an indexed version of Ident was inferred: $Ident_i$. All soft-inconsistent words for that pair were then given the index of the newly minted constraint (in this case, *i*), and the tableau was updated to include the violation profile of the indexed constraint. A summary of this entire procedure is given in (9):

[2] If the relevant indexed constraint did already exist, but there were soft-inconsistent words for this constraint pair that were not yet associated with this indexed constraint, then the index of this relevant indexed constraint was added to these newly discovered exceptional words.

(9) *Summary of the current implementation of the soft inconsistency proposal*
Start with P(X>>Y) = P(Y>>X) = 0.5 for each constraint pair {X,Y}
Repeat the following procedure until likelihood of the training data is at least 95%:

1. For each constraint pair {X,Y}:
   - (Re-)Estimate P(X>>Y) and P(Y>>X) for each individual word and P(X>>Y) and P(Y>>X) for the entire lexicon (see Appendix)
   - **Apply soft inconsistency diagnostic in (8)**: find which words are soft-inconsistent w.r.t. this constraint pair and the lexicon
2. Find which constraint pair has the greatest summed ranking tendency divergence between its soft-inconsistent words ($W_{inc(XY)}$) and the overall lexicon

$$\underset{\{X,Y\}}{\operatorname{argmax}} \sum_{w_{inc(XY)} \in W_{inc(XY)}} \left| P(X \gg Y)_{w_{inc(XY)}} - P(X \gg Y)_{lexicon} \right|$$

3. Given this "most exceptional constraint pair": if appropriate, add to the constraint set an indexed version of whichever constraint in this pair is preferred to be on top by the soft-inconsistent words; mark these soft-inconsistent words with the corresponding index

5 Case study: Hebrew spirantization

5.1 *Data and simulation setup* The specific implementation of the soft-inconsistency proposal as laid out in section 4 was tested on simplified data on Hebrew spirantization (Temkin-Martínez 2010), as previewed briefly in (2c) in section 1. These data contain the generalization that postvocalic /p,b,k/ optionally spirantize to [f,v,χ] in default words, while in exceptional words, they either obligatorily spirantize, or never spirantize at all. It is this generalization that I will be modeling in the case study. The specific dataset that was used for this case study is as follows:

(10) *Simplified Hebrew dataset used for the case study*

postvocalic underlying stops			*non-postvocalic underlying stops*
/me**k**ase/	→ [me**k**ase ~ me**χ**ase]		/lin**p**oʃ/ → [lin**p**oʃ]
/ʃa**b**ar/	→ [ʃa**b**ar ~ ʃa**v**ar]		/lis**b**ol/ → [lis**b**ol]
/da**k**ar/	→ [da**k**ar]	exceptions	/liʃ**k**oa/ → [liʃ**k**oa]
/me**b**arer/	→ [me**v**arer]		/liʃ**p**oχ/ → [liʃ**p**oχ]
/me**b**atel/	→ [me**v**atel]		/liz**k**ot/ → [liz**k**ot]
/ga**b**a/	→ [ga**v**a]		

This dataset was generated from the judgment experiment corpus in Temkin-Martínez (2010). All stems with a single (non-final) underlying stop were extracted, and the relative frequencies of words with postvocalic and non-postvocalic optional spirantization, obligatory spirantization (= spirantization ≥90% of the time), and no spirantization (= spirantization ≤10% of the time) were computed. In the simplified dataset, the proportions of these 6 types were retained, but all non-postvocalic within-word variation was leveled, in order to focus on the postvocalic variation.

The original Hebrew data in Temkin-Martínez (2010) necessitate the use of diacritics (at least in a categorical representational context) because of the following four-way contrast in non-postvocalic context: stop only, e.g., [b]; within-word variation between stop and fricative with a preference for the stop, e.g., [b > v]; within-word variation between stop and fricative with a preference for the fricative, e.g., [v > b]; fricative only, e.g., [v]. While a three-way contrast could be represented with underspecification (cf. Inkelas et al. 1997) (/-continuant/ for [b], /0 continuant/ for [b ~ v], and [+continuant] for [v]), a four-way contrast like the one found in Hebrew cannot be easily mapped onto an underspecification pattern.

In the data used in this case study, all relative frequencies of within-word variants were leveled. That is, within-word variation was presented to the learner as a disjunction between two candidates, and a match between data and grammar was assessed whenever the grammar predicted either of the two attested candidates.

The constraints used in the simulation were very simple, and were based on the Stochastic OT analysis

in Temkin-Martínez (2010). The three Markedness constraints were *Stop/V_ (one violation for vowel-stop sequences), *Stop (one violation for any stop), and *NonSibilantFricative (one violation for any non-sibilant fricative like [f], [v], or [χ]). The Faithfulness constraints used were Ident and Max, since the deviations form underlying forms considered here were segment change and segment deletion. In forms with postvocalic /p,b,k/, spirantization was considered, as well as deletion of the preceding vowel (which is another strategy to avoid violations of *Stop/V_): an example candidate set is shown in (11a). In forms with non-postvocalic /p,b,k/, the unfaithful mappings considered also included spirantization, but featured deletion of the preceding consonant instead of deletion of a vowel – as exemplified in (11b).

(11) a. *example of postvocalic /p,b,k/ candidate set*
/mekase/: [mekase, meχase, mkase, mχase]

b. *example non-postvocalic /p,b,k/ candidate set*
/linpoʃ/: [linpoʃ, linfoʃ, lipoʃ, lifoʃ]

5.2 *Results* Within 100 runs of up to 80 iterations with the learner described in section 4, and with the data and setup described in section 5.1, the learner always reached its goal (≥95% training data likelihood) within 16–34 iterations (average: 20.7). Learning performance was evaluated on whether the analysis in Temkin-Martínez (2010) was picked up by the learner. Specifically, I investigated whether the (simplified) default pattern was picked up by the learner (i.e., whether removing the indices from the inputs in the final grammars yielded variable spirantization after a vowel – and no spirantization and no deletion in non-postvocalic contexts), and whether diacritics/indiced were assigned to exceptions only (I looked both at "false alarms" – non-exceptions that were falsely hypothesized to be exceptions – and undetected exceptions). The results of these tests, averaged between runs, are given in (12):

(12)

Performance		*Default pattern: correct range of variation?*	*Diacritic/index assigned*
Non-exceptions	*Post-V*	No deletion, variable spirantization: **99.1%**	**0%**
	Post-C	No deletion of spirantization: **97.2%**	**1.8%**
Exceptions	*Post-V*	–	**87.8%** (/b/ → [v]: 100%; /k/ → [k]: 51%)

As can be seen in (12), performance on the default pattern is very good (98.1% across all contexts): the learners were able to extract a grammar that reflects the intended default pattern, even if most of the post-vocalic /p,b,k/ words were exceptions. As for assignment of indices to inputs, "false alarms" were very infrequent (1.4% occurrence across all contexts), while exceptions were detected almost 90% of the time. Interestingly, exceptions with obligatory spirantization (/me**b**arer/ → [me**v**arer], /me**b**atel/ → [me**v**atel], /ga**b**a/ → [ga**v**a], indicated in (12) as /b/ → [v]) were detected and accounted for 100% of the time, while the exception with lack of spirantization (/da**k**ar/ → [da**k**ar], indicated win (12) as /k/ → [k]) were only detected in about 50% of all cases.

The underperformance of the learner on recognizing /da**k**ar/ → [da**k**ar] can be understood from a combination of factors. The tendency towards spirantization in the dataset is stronger than the tendency towards non-spirantization (if non-postvocalic cases are taken into consideration), so that an exception that goes against the spirantization tendency is more difficult to detect. In addition, there is only one exception of this kind in the dataset, allowing it to "fly under the radar". A learner with soft inconsistency detection does predict that exceptions to less prominent generalizations will be more difficult to learn, since these generalizations will have more trouble meeting the second clause in (5b) in section 3 or (8) in section 4.2. Classes of exceptions with fewer members will also have a higher tendency to remain unnoticed, since there is a chance of non-detection (because of faulty pairwise ranking probability estimation) at every instance of soft inconsistency detection, and classes of exceptions with fewer members simply have fewer instances of soft inconcistency detection applied to them. For these reasons, a single-member exception class that goes against a less prominent pattern in the dataset, like /dakar/ in the current simpified daaset, is predicted to become default-conforming over time, changing to /da**k**ar/ → [da**k**ar ~ da**χ**ar]. However, because the actual Hebrew dataset presented in Temkin-Martínez (2010) is more complex, the specific predictions for actual Hebrew are less clear.

6 Concluding remarks

In this paper, I have presented a proposal for learning OT grammars (Prince and Smolensky 1993/2004) for datasets that involve both within-word and between-word variation (Coetzee and Pater 2011). There are previous proposals for learning within-word variation in OT (including Boersma 1998, Jarosz 2015), as well as for learning between-word variation in OT (Pater 2010, Becker 2009, Coetzee 2009). As shown in section 2.2, these two types of proposals are not mutually compatible, since they have mutually incompatible responses to opposite ranking requirements within the dataset.

The current proposal combines the two approaches into one, by making the probabilistic learners (in particular, the one proposed by Jarosz 2015) sensitive to the distinction between word-types and word-tokens, and by adapting to a probabilistic learning framework the inconsistency criterion developed by Pater (2010), Becker (2009), and Coetzee (2009) for a categorical learner (Tesar 1995). The result is what I call here *soft inconsistency*: a criterion for inferring the existence of between-word variation from opposite ranking tendencies between words, as computed through a probabilistic ranking learner. In section 4, a specific implementation of the proposal is given in Expectation Driven Learning (Jarosz 2015) to learn indexed constraint grammars (Kraska-Szlenk 1995, Pater 2000, 2010).

When applied to a simplified version of Hebrew optional spirantization (Temkin-Martínez 2010; see section 5.1 for the simplified variant), this learner performed exceptionally well at finding the default pattern (about 98% accuracy), and at keeping non-exceptional words from being assigned an exceptionality diacritic (about 99% accuracy). Exceptional words were assigned an exceptionality diacritic about 88% of the time, with the three exceptional words that had obligatory spirantization (and went against the most prominent pattern in the dataset) being assigned a diacritic 100% of the time, while the one exceptional word that had no spirantization (and went against a less prominent pattern in the dataset) being assigned a diacritic only 51% of the time.

In the future, this slight underprediction of exceptionality on the part of the learner should be investigated in more case studies, especially from the point of view of historical change: will certain types of phonological exceptionality be more likely to disappear, as predicted by the current model?[3] In general, more case studies should be attempted in order to assess more clearly this model's success at discovering phonological exceptionality in the face of within-word variation. Crucially, the model should also be tested on an entire dataset used in an experiment (for instance, the entire dataset from Temkin-Martínez 2010), so that the predictions of the model can be matched to the results obtained in the experiment. Another important direction for future work will be to try various implementations of the soft inconsistency criterion, both in different theoretical frameworks (e.g., in Cophonology Theory, e.g., Inkelas 1998, instead of Indexed Constraint Theory) and in different learning frameworks (e.g., in Stochastic OT, Boersma 1998, instead of Expectation Driven Learning). This is crucial to be able to tell apart the effects of implementation and the effects of the soft inconsistency criterion.

Finally, an important question is posed by how grammars with indexed constraints generalize. Earlier results such as Hayes et al. (2009) show that when presented with nonce words, speakers extend between-word variation onto these words despite the fact that they have no lexical marking for exceptionality. One model for such behavior is provided by Becker and Gouskova (2016) – I hope to be able to combine the current soft inconsistency proposal with some form of this model in the future in order to investigate how speakers learn to make judgments about between-word variation in nonce words as well as real words.

7 Appendix

7.1 *Sampling procedure for Pairwise Ranking Grammars (Jarosz 2015)* The sampling procedure for Pairwise Ranking Grammars is meant to constrain the pairwise ranking probability model from generating logically impossible constraint rankings (see section 2.1). The procedure starts with a Pairwise Ranking Grammar, as in (13), and concludes with a full ranking of the constraints in the grammar.

[3] Many thanks to Patrycja Strycharczuk for bringing up this point.

(13) *A sample Pairwise Ranking Grammar*

P(*Stop >> Ident) =	0.20	P(Ident >> *Stop) =	0.80
P(*Stop >> *Stop/V_) =	0.01	P(*Stop/V_ >> *Stop) =	0.99
P(*Stop/V_ >> Ident) =	0.70	P(Ident >> *Stop/V_) =	0.30

The sampling procedure is stepwise, and works with conditional probabilities. If a possible new pairwise ranking is logically independent from the ranking established so far (for instance, *Stop >> Ident given *Stop >> *Stop/V_), the potential new ranking's conditional probability will be the same as their non-conditional probability. However, if a possible new pairwise ranking is logically dependent on the ranking established so far (for instance *Stop >> Ident given *Stop >> *Stop/V_ >> Ident), its conditional probability will be 0 or 1, depending on whether it is entailed by or inconsistent with the ranking established so far. The way that Jarosz's (2015) sampling procedure uses conditional probabilities can be summarized as in (14); Jarosz 2015 gives a more operationalized formulation, but (14) is logically consistent with it. In (15), I illustrate how (14) would be applied to the PRG in (13).

(14) *Summary of sampling procedure*

Create a random order of all logically possible pairwise constraint rankings
Initialize the full ranking, *F*, with the empty ranking [ ]
Going through the possible pairwise rankings one at a time, repeat the following procedure:
  Choose the current possible pairwise ranking, *X* >> *Y*
  Compute the probability of *X* >> *Y* given the full ranking (*F*) so far
    P(*X*>>*Y* | *F*)
  Flip a coin weighted according to P(*X*>>*Y* | *F*) – this returns True or False
    If True, add *X* >> *Y* to the full ranking *F*; If False, add *Y* >> *X* to the full ranking *F*

(15) *Example of applying the sampling procedure in (14) to the PRG in (13)*

Create a random order: 1. Ident >> *Stop/V_ | 2. Ident >> *Stop | 3. *Stop/V_ >> *Stop | ... | 6. ...
F so far: [ ] P(Ident >> *Stop/V_ | *F*) = P(Ident >> *Stop/V_) = 0.3
Sample True
F so far: [Ident >> *Stop/V_] P(Ident >> *Stop | *F*) = P(Ident >> *Stop) = 0.8
Sample False
F so far: [*Stop >> Ident >> *Stop/V_] P(*Stop/V_ >> *Stop | *F*) = 0 *by ranking transitivity*
Sample False *etc.*

7.2 *A summary of Expectation Driven Learning (Jarosz 2015)* Expectation Driven Learning (EDL) is a learning procedure for Pairwise Ranking Grammars (PRGs). EDL exploits the variance generated by this sampling procedure and the conditional probabilities in the sampling procedure to estimates ranking probabilities using the general framework of Expectation Maximization (Dempster et al. 1977). The learning procedure can be summarized as in (16):

(16) *Pseudo-code for EDL procedure*

Initialize the PRG with initial probabilities (in the current case, with all probabilities being 0.5)
Iterate the following procedure until convergence:
  For each constraint pair {X,Y}:
    For each $word_n$ in the data:
      Sample *r* (here, *r* = 50) rankings from the PRG initializing *F* at X >> Y
        match(X>>Y) = how many of these yield an attested output as the winning candidate
      Sample *r* rankings from the PRG initializing *F* at Y >> X
        match(Y>>X) = how many of these yield an attested output as the winning candidate
      Estimate per-word pairwise ranking probabilities (see Jarosz 2015 for explanation)

$$P(X \gg Y)_{word_n\text{-}new} = \frac{match(X \gg Y) \times P(X \gg Y)_{PRG}}{match(X \gg Y) \times P(X \gg Y)_{PRG} + match(Y \gg X) \times P(Y \gg X)_{PRG}}$$

$$P(Y \gg X)_{word_n\text{-}new} = 1 - P(X \gg Y)_{word_n\text{-}new}$$

Estimate lexicon-wide pairwise ranking probabilities

$$P(X \gg Y)_{lexicon} = \frac{\Sigma_{word} P(X \gg Y)_{word}}{\Sigma_{word} P(X \gg Y)_{word} + \Sigma_{word} P(Y \gg X)_{word}}$$

Update grammar: copy lexicon-wide pairwise ranking probability estimates into PRG

References

Anttila, Arto. 1997. "Deriving Variation from Grammar." In *Variation, Change, and Phonological Theory*, ed. by Frans Hinskens, Roeland van Hout, and W. Leo Wetzels. 35–68. Amsterdam: John Benjamins.

Anttila, Arto. 2002. "Morphologically Conditioned Phonological Alternations." *Natural Language & Linguistic Theory* 20(1).1–42.

Boersma, Paul. 1998. *Functional Phonology: Formalizing the Interactions between Articulatory and Perceptual Drives*. Amsterdam: University of Amsterdam dissertation.

Becker, Michael. 2009. *Phonological Trends in the Lexicon: The Role of Constraints*. Amherst, MA: University of Massachusetts Amherst dissertation.

Coetzee, Andries W. 2009. "Learning Lexical Indexation." *Phonology* 26(1).109–45.

Coetzee, Andries W, and Joe Pater. 2011. "The Place of Variation in Phonological Theory." *Handbook of Phonological Theory*, 2nd edition, ed. by John A. Goldsmith, Jason Riggle, and Alan C. Yu, 401–34. Hoboken, NJ: Wiley-Blackwell.

Dempster, Arthur P., Nan M. Laird, and Donald B. Rubin. 1977. "Maximum Likelihood from Incomplete Data via the EM Algorithm." *Journal of the Royal Statistical Society. Series B (Methodological)* 39(1).1–38.

Goldwater, Sharon, and Mark Johnson. 2003. "Learning OT Constraint Rankings Using a Maximum Entropy Model." *Proceedings of the Stockholm Workshop on Variation within Optimality Theory*, ed. by Jennifer Spenader, Anders Eriksson, and Östen Dahl, 111–20. Stockholm: Linguistics Department, Stockholm University.

Hayes, Bruce, Kie Zuraw, Péter Siptár, and Zsuzsa Londe. 2009. "Natural and Unnatural Constraints in Hungarian Vowel Harmony." *Language* 85(4).822–63.

Inkelas, Sharon. 1998. "The Theoretical Status of Morphologically Conditioned Phonology: A Case Study from Dominance." *Yearbook of Morphology 1997*. 121–55.

Inkelas, Sharon, Orhan Orgun, and Cheryl Zoll. 1997. "The implications of lexical exceptions for the nature of grammar." *Constraints and Derivations in Phonology*, ed. by Iggy Roca. 393–418. Oxford: Clarendon Press.

Jarosz, Gaja. 2015. *Expectation Driven Learning of Phonology*. Ms., University of Massachusetts Amherst.

Kraska–Szlenk, Iwona. 1995. *The Phonology of Stress in Polish*. Urbana-Champaign, IL: University of Illinois, Urbana-Champaign dissertation.

Manning, Christopher D., and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing*. Cambridge, Massachusetts: MIT Press.

Moore-Cantwell, Claire. 2017. "Concurrent learning of the lexicon and phonology." Paper presented at the *LSA 2017 Annual Meeting*.

Nagy, Naomi, and William Reynolds. 1997. "Optimality Theory and variable word-final deletion in Faetar." *Language Variation and Change* 9.37–56.

Pater, Joe. 2000. "Non-Uniformity in English Secondary Stress: The Role of Ranked and Lexically Specific Constraints." *Phonology* 17(2).237–74.

Pater, Joe. 2010. "Morpheme-Specific Phonology: Constraint Indexation and Inconsistency Resolution." *Phonological Argumentation: Essays on Evidence and Motivation*, ed. by Steve Parker. 123–54. London: Equinox.

Potts, Christopher, Joe Pater, Karen Jesney, Rajesh Bhatt, and Michael Becker. 2010. "Harmonic Grammar with Linear Programming: From Linear Systems to Linguistic Typology." *Phonology* 27(1).1–41.

Prince, Alan S., and Paul Smolensky. 1993/2004. *Optimality Theory: Constraint Interaction in Generative Grammar*. Malden, MA: Blackwell.

Smolensky, Paul, Matt Goldrick, and Donald Mathis. 2014. "Optimization and Quantization in Gradient Symbol Systems: A Framework for Integrating the Continuous and the Discrete in Cognition." *Cognitive Science* 38(6).1102–38.

Temkin–Martínez, Michal. 2010. "Sources of Non-Conformity in Phonology: Variation and exceptionality in Modern Hebrew Spirantization." Los Angeles, CA: University of Southern California dissertation.

Tesar, Bruce. 1995. "Computational Optimality Theory." Denver, CO: University of Colorado dissertation.