Convergence Properties of a Gradual Learning Algorithm for Harmonic Grammar*

Paul Boersma and Joe Pater, May 21, 2008

Abstract. This paper investigates a gradual on-line learning algorithm for Harmonic Grammar. By adapting existing convergence proofs for perceptrons, we show that for any nonvarying target language, Harmonic-Grammar learners are guaranteed to converge to an appropriate grammar, if they receive complete information about the structure of the learning data. We also prove convergence when the learner incorporates evaluation noise, as in Stochastic Optimality Theory. Computational tests of the algorithm show that it converges quickly. When learners receive *incomplete* information (e.g. some structure remains hidden), tests indicate that the algorithm is more likely to converge than two comparable Optimality-Theoretic learning algorithms.

Keywords: learnability, Optimality Theory, Harmonic Grammar, Stochastic OT, Noisy HG, perceptron

1 Introduction

An attractive feature of Optimality Theory (OT; Prince and Smolensky 1993/2004) is the existence of associated *learning algorithms*. In general, a learning algorithm works as follows: given a finite set of data from some *target language*, the algorithm will try to find a grammar that is consistent with that dataset; if the dataset is *sufficiently rich*, i.e. if it contains enough information about all aspects of the target language, the algorithm can hope to find a grammar that is appropriate for the whole target language. In OT, a language is a potentially infinite set of pairs of *input* and *output* forms, and a grammar is a set of *ranked constraints*. A learning algorithm for OT is therefore given a finite set of input-output pairs and tries to find a constraint ranking that renders all these pairs *optimal*, i.e. a ranking such that for each given input, the optimal candidate in that input's candidate set equals the given output; if the set of given input-output pairs is sufficiently rich, the resulting grammar can be appropriate for the whole target language.

Prince and Smolensky consider only *nonvarying languages*, i.e. languages in which every input form has a unique optimal output form. Prince and Smolensky implement this uniqueness by proposing that constraints are ordered in a *totally ranked hierarchy*, in which equal ranking between constraints is banned, and also that the constraint ranking does not vary from evaluation to evaluation. For this original version of OT, Tesar (1995) devised a family of learning algorithms known as Constraint Demotion (CD). These algorithms are provably *convergent*, i.e., when

* Thanks to Rajesh Bhatt, Michael Collins, Karen Jesney, Kie Zuraw, and participants in Linguistics 794, Fall 2007 UMass for comments on earlier drafts. Thanks also to Mark Johnson, Giorgio Magri, Patrick Pratt and Colin Wilson for useful discussion, and to Bruce Tesar for supplying a list with all the overt forms of all the languages needed for the RIP simulations.

given a sufficiently rich set of data, they are guaranteed to find at least one totally ranked hierarchy appropriate for the target language.[1]

However, there are natural language phenomena that have been claimed to involve multiple optimal outputs for a single input: so-called free variation. In OT, such *varying languages* have been analyzed in terms of random variation between whole grammars (Kiparsky 1993), floating constraints *within* a grammar (Reynolds 1994, Nagy and Reynolds 1997), partial ranking (Anttila 1997), and noisy ranking (Boersma 1997, Boersma and Hayes 2001); for overviews, see Anttila 2007 and Coetzee and Pater to appear. In these views of variation, the constraint ranking is in a total order during every evaluation: each time the grammar is given an input, it produces a single optimal output. However, random variation in the ordering of the constraints across instances of evaluation can produce variation in outcomes.

Of these varieties of OT that can represent variation, Stochastic OT (Boersma 1997, Boersma and Hayes 2001) appears to be the only one that has an associated learning algorithm.[2] To deal with variable outcomes, Stochastic OT incorporates *evaluation noise*: the rankings of the constraints have values along a continuous numerical scale and are subject to random variation each time the grammar is used to evaluate a candidate set; the total ranking that is derived from this can therefore vary across evaluations. The associated Gradual Learning Algorithm (henceforth, *OT-GLA*) usually succeeds in finding a grammar that reproduces probability distributions over outputs in the learning data, if those distributions can be represented by Stochastic OT. As well as being able to learn languages with variation, OT-GLA differs from CD in its robustness in the face of a moderate number of mistakes in the learning data, and in exhibiting gradual learning curves similar to those observed in natural language acquisition (Boersma 1997, Boersma and Levelt 2000). Both of these attributes are desirable if the learning theory is to function as a realistic model of language acquisition. However, OT-GLA also diverges from CD in that it is not guaranteed to converge: there exist nonvarying languages that can be represented by OT (and therefore by Stochastic OT) but cannot be learned by OT-GLA (Pater 2008).

In this paper we present an approach to grammar learning that combines the greatest strengths of the two OT learning algorithms. Like CD, the learning algorithm is guaranteed to converge on an appropriate grammar (if there is one) for any nonvarying set of learning data. Like OT-GLA, the learning algorithm is compatible with a theory of grammar that represents variation, and when combined with that grammar model, it produces gradual learning curves and is robust in the presence of mistakes in the learning data.

We obtain this combination of advantages by stepping outside the framework of OT and instead adopting the framework of *Harmonic Grammar* (HG; Legendre, Miyata and Smolensky 1990, Smolensky and Legendre 2006), in which constraints

[1] There is one exception: the most commonly used of these algorithms, EDCD, is not guaranteed to converge to a totally ranked hierarchy. Fortunately, it can easily be repaired to do so (Boersma 2008). All EDCD simulations in the present paper are performed with the repaired version. See also footnotes 9 and 15.

[2] The Constraint Demotion algorithms were not designed for handling free variation, and must therefore naturally be expected to fail to converge when fed with data from a varying language, as they indeed turn out to do (Tesar 1995: 103, Tesar and Smolensky 1998: 249–250, Boersma and Hayes 2001).

are numerically weighted, rather than ranked as in (nonstochastic and stochastic) OT. To deal with variable outcomes, however, we adopt Stochastic OT's evaluation noise: the constraint weights are subject to random variation across evaluations. We call the resulting grammar model *Noisy HG* (not "Stochastic HG", because Maximum Entropy constraint grammars such as proposed by Johnson 2002 form another stochastic version of HG).

In the present paper we restrict ourselves to the learning of nonvarying languages. However, the learners that we consider can be either *noisy HG learners*, who incorporate a substantial evaluation noise during learning, or *nonnoisy HG learners*, for whom the evaluation noise is zero. For both types of learners, we call the associated learning algorithm *HG-GLA* (for Harmonic Grammar's Gradual Learning Algorithm). This algorithm first appeared, in the guise of a connectionist network, in Soderstrom, Mathis and Smolensky 2006, and an identical algorithm was applied to Maximum Entropy constraint grammars by Jäger (2003/to appear). For noisy learners, the algorithm was implemented in the Praat program in April 2007 and has been investigated by Jesney (2007), Boersma and Escudero (in press), Pater (2008), Jesney and Tessier (2007a), Pater, Bhatt and Potts (2007), and Coetzee and Pater (to appear).

In section 2 we introduce the theory of HG grammars and learning. In sections 3 and 4 we provide the core results of the paper: proofs of convergence and computational tests. In section 3 we first prove that nonnoisy HG learners are guaranteed to succeed in finding appropriate constraint weights for any nonvarying finite set of language data generated by HG. We then provide results from computer simulations on randomly generated nonvarying languages, which show that in practice HG-GLA converges very fast, and also that it converges even if constraint weights are limited to positive values in a framework we call Exponential HG. In section 4 we prove that noisy learners can learn nonvarying languages just as well as the nonnoisy learners of section 3; again, computer simulations show that these learners tend to be fast and that the learning result also holds with Exponential HG. Furthermore, individual learners display a natural-looking learning curve. As a comparison, we show that similar simulations with a Stochastic OT learner with OT-GLA display a moderate incidence of non-convergence.

Section 5 briefly discusses how HG learners generalize from finite datasets to infinite target languages, and compares the nature of generalization in OT and HG.

In sections 6 and 7, we present an initial investigation and discussion of the behavior of HG-GLA on learning problems that present well-known challenges to CD, as well as to other learning algorithms. Here we compare several learning algorithms, restricting ourselves to algorithms that are both *on-line* and *error-driven*. On-line algorithms receive a single datum at a time, which is processed and learned from, and then forgotten. Error-driven algorithms change their grammar hypothesis only if an incoming datum is ungrammatical in their current state of the grammar (Wexler and Culicover 1980: 127). The on-line error-driven version of CD is called Error-Driven Constraint Demotion (EDCD; Tesar 1995); both OT-GLA and HG-GLA have always been applied as on-line error-driven learning algorithms.

In section 6, we discuss the *subset problem* for HG and OT phonological acquisition (Smolensky 1996). When a learner is presented only with non-alternating

phonological forms, its final grammar may come to accept forms that an analyst, or human learner, would judge not to belong to the target language. We draw on previous research (Boersma and Levelt 2003, Jesney and Tessier 2007a) to show that in several scenarios, HG-GLA avoids subset traps that foil EDCD and/or OT-GLA.

In section 7, we discuss the *hidden structure* problem. In human language acquisition, much linguistic structure cannot be directly inferred from the observed learning data. If this structure is not provided to a learner, the data may be insufficiently rich to guarantee convergence. Tesar and Smolensky (2000) propose an approach to the learning of such hidden structure termed Robust Interpretive Parsing (RIP), and test this approach in conjunction with EDCD on a representative hidden structure problem involving metrical structure. We present results on the same learning problem using an implementation of RIP with three learning algorithms (EDCD, OT-GLA, HG-GLA) and six grammar models (nonstochastic and stochastic OT, nonnoisy and noisy HG, and nonnoisy and noisy Exponential HG). It turns out that HG learners have a larger chance of converging than OT learners, with Noisy HG learners having the largest chance.

Section 8 concludes that HG has a learning algorithm that in all respects investigated either equals or outperforms the OT learning algorithms that we have compared it to. We regard these learning results as providing evidence that HG deserves to be reconsidered as a model of generative grammar (as also argued by Pater, Bhatt and Potts 2007 and others cited there, as well as by Keller and Asudeh 2002[3]).

2 Harmonic Grammar and a Gradual Learning Algorithm

We assume that our readers are familiar with OT as presented in Prince and Smolensky 1993/2004 as well as with CD (see Kager 1999, Boersma, Dekkers and Van de Weijer 2000, and McCarthy 2002, 2008 for introductions to OT and CD). Because HG is likely less familiar, we first provide an introduction to this theory of generative grammar, and to its noisy version, along with the associated learning algorithm.

In HG, the well-formedness of a linguistic representation is defined in terms of a *harmony* function (Legendre, Miyata and Smolensky 1990, Smolensky and Legendre 2006). The harmony of a representation is a weighted sum of its constraint violations and satisfactions. Prince and Smolensky (1993/2004: 236) point out that the optimal member of a candidate set can be defined in HG terms. In (1), we provide a simple example of an HG candidate evaluation. Following Legendre, Sorace and Smolensky (2006) we regard constraint violations as negative constraint satisfactions and therefore denote them by negative numbers in tableau cells (an empty cell means no violation); the weights of the constraints are shown beneath the constraint names, and harmony values are shown at the end of the candidate rows. The candidate (i_1, o_{11}) incurs one violation of C_1, which has a weight of 1.0, so the harmony of this candidate is $1.0 \cdot (-1) = -1$. Because the second candidate (i_1, o_{12}) violates a constraint with

[3] Keller and Asudeh (2002) argue for HG on the basis of cumulative constraint interaction, which cannot be represented by ranked constraints. They mislabel this as an argument against OT-GLA; it is in fact an argument against OT.

lower weight, it has a higher harmony than the first candidate, namely −0.9. The last candidate (i_1, o_{13}) has two violations of C_2 and one of C_1, which yields a harmony of $1.0\cdot(-1) + 0.9\cdot(-2) = -2.8$, which is also lower than −0.9. Following OT terminology, the second candidate is therefore deemed *optimal* (= maximally harmonic), as indicated by the pointing finger in (1).

(1) *Optimality as maximal HG harmony*

i_1	C_1 1.0	C_2 0.9	
o_{11}	−1		−1.0
☞ o_{12}		−1	−0.9
o_{13}	−1	−2	−2.8

The error-driven learning algorithm we adopt is very similar to OT-GLA (Boersma 1997). The learner is given a single input-output pair at a time, and adjusts the constraint weights only when this pair is not optimal under the learner's current grammar hypothesis. Error-driven learning is known from perceptrons (Rosenblatt 1958) and their connectionist relatives, from theories of language acquisition (Wexler and Culicover 1980: 127), from EDCD (Tesar 1995), and from parameter setting algorithms (Gibson and Wexler 1994, Fodor 1998, Yang 2002).

The following is a simple example of an "error" made by an HG learner. Suppose that at some point during the acquisition period a learner has the weights shown in (1), and that the learner is then presented with the learning datum (i_1, o_{11}). The learner will consider o_{11} as the "correct" output for the input i_1, as indicated with the check mark in tableau (2). Meanwhile, the learners' own optimal output (the one with highest harmony given her current grammar) is o_{12}, and this is indicated with the pointing finger in (2); the learner will now consider this form "incorrect".

(2) *An "error" in HG*

i_1	C_1 1.0	C_2 0.9	
√ o_{11}	−1		−1.0
☞ o_{12}		−1	−0.9
o_{13}	−1	−2	−2.8

Upon making such an error, the learner proceeds to update the constraint weights so as to render the error less likely when encountering future data with similar properties. The update procedure we use is one that is widely used in connectionist and statistical learning, and has been applied to grammatical constraint weighting under several names. Under the name of *stochastic gradient ascent*, Jäger (2003/to appear) applied it to HG's stochastic cousin called *Maximum Entropy grammar*; under the name of *gradient descent in error*, Soderstrom, Mathis and Smolensky (2006: eqs. 14, 18, 21, 35d) applied it to a connectionist implementation of HG; and under the name of

perceptron update rule Pater (2008) applied it to the present version of (nonnoisy) HG. Here we refer to it by its function, namely *HG-GLA*: Harmonic Grammar's gradual learning algorithm.

The first step of the learning algorithm is to calculate the difference between the constraint satisfactions of the "incorrect" and "correct" forms. This can be represented as an *error vector*, as in (3), which results from subtracting (in (2)) the scores of the incorrectly optimal (i_1, o_{12}) from the scores of the correct (i_1, o_{11}); for instance, for C_2 we compute 0 minus −1, which yields +1. The representation in (3) is the HG equivalent of the mark-data pair or winner-loser pair used in OT learning (Tesar 1995 *et seq*.), and was first explicitly discussed for Maximum Entropy grammars by Goldwater and Johnson (2003) and for HG by Becker and Pater (2007).

(3) *The error vector: scores of correct pair minus scores of incorrect pair*

Correct ~ incorrect	C_1	C_2
(i_1, o_{11}) ~ (i_1, o_{12})	−1	+1

The next step is that the constraint weights are updated by adding to them the values of this error vector, each scaled (multiplied) by a small constant ε, termed *plasticity* in the GLA literature (or *learning rate* in the neural network and statistical learning literature). As a result, the weight of a constraint that is violated more in the correct form than in the incorrect form (here, C_1) is lowered, and the weight of a constraint that is violated more in the incorrect form than in the correct form (here, C_2) is raised. If we set the plasticity at ε = 0.4, and add the scaled error values to the constraint weights, we obtain tableau (4).

(4) *Updated grammar*

i_1	C_1 0.6	C_2 1.3	
√ ☞ o_{11}	−1		−0.6
o_{12}		−1	−1.3
o_{13}	−1	−2	−3.2

In tableau (4), 0.4·(−1) = −0.4 has been added to the weight of C_1, and 0.4·(+1) = 0.4 has been added to the weight of C_2. As a result, the grammar has changed so much that the behavior of the learner when presented with the next instance of i_1 will have changed: as we see in (4), o_{11} has become the optimal form in the learner's grammar, and since this form is also the correct form, learning has stopped as far as input i_1 is concerned. With plasticities smaller than 0.4, the acquisition period is likely to take longer.

The most accurate name for this type of learning algorithm (starting with the perceptron) is *stochastic gradient descent in error*: "stochastic" because it is an on-line learning algorithm that relies on a randomized sequence of on-line learning data, and "gradient descent in error" because the "error", which is the harmony difference

between the learner's incorrect optimum and the correct form, is reduced[4]; "in error" also implies the error-drivenness of the algorithm, in contrast with algorithms that update their parameters on every input regardless of correctness.

As section 3 shows, this learning algorithm is quite effective: given any finite dataset from a nonvarying language that can be described by an HG grammar, it is guaranteed to converge on a correct weighting. Like the proofs of convergence for CD, the proof in section 3 is only valid for nonvarying languages.

As mentioned in the introduction, it has been claimed that natural languages sometimes show "free" variation. In phonology, this is the variation between different surface representations for a single underlying representation. To deal with such cases, we modify HG by incorporating *evaluation noise*, as in Stochastic OT (Boersma 1997), so that we obtain *Noisy HG*.[5] In Noisy HG, every time the grammar is used to evaluate a candidate set, the weighting values are perturbed by noise: each constraint's value is temporarily altered by adding a random positive or negative number sampled from a normal distribution with zero mean. This will lead to different weightings across instances of evaluation, which will produce variation. The proportion of times a candidate in any given candidate set is chosen is thus determined by the weights of the constraints and by the width of the noise distribution. The effect of noisy evaluation is illustrated in (5), which shows two evaluations for the input i_2 in the same noisy grammar. The basic weights (corresponding to the *ranking values* of Stochastic OT) of the constraints are 1.1 and 1.0 for C_1 and C_2 respectively. The weights after the addition of noise (corresponding to the *disharmonies* of Stochastic OT) are shown in parentheses. Depending on these stochastic weights, either output o_{22} or output o_{21} will be chosen, as these two tableaux show. Because o_{22} violates the constraint with a lower basic weight, it will be optimal in a greater proportion of evaluations than o_{21}.

(5) *Variation in choices of optima in Noisy HG: one grammar, two outputs*

i_2	C_1 1.1 (1.1)	C_2 1.0 (0.9)	
o_{21}	–1		–1.1
☞ o_{22}		–1	–0.9

i_2	C_1 1.1 (1.0)	C_2 1.0 (1.1)	
☞ o_{21}	–1		–1.0
o_{22}		–1	–1.1

As we show in section 4, adding such noise does not affect the ability of the learner to find correct weights for nonvarying languages, and it does lead to realistic learning curves.

[4] Technically, the error vector in (3) is the "gradient" of the error (harmony difference) in constraint weight space.

[5] Another stochastic variant of HG is *Maximum Entropy grammar* (Johnson 2002, Goldwater and Johnson 2003, Jäger 2003/to appear, Fischer 2005, Wilson 2006). In this model, every output candidate has a relative probability that rises exponentially with its harmony. One difference from Noisy HG is that Maximum Entropy grammars always allow harmonically bounded candidates to surface (Jäger and Rosenbach 2006), whereas in Noisy HG such candidates have zero probability if the weights (including the evaluation noise) are limited to positive values (Jesney 2007; see here §3.5). Yet another stochastic variant of HG is Goldrick and Daland's (2007) "disrupted Hopfield network", which is essentially a notational variant of Noisy HG, so that our proof in §3.3 is applicable to it.

Before we turn to our proof, we provide one slightly more complex illustrative example to more fully demonstrate HG evaluation and learning. The OT literature usually speaks of constraint *violations* rather than of constraint *satisfactions*, but the OT formalism does not really disallow the use of positive constraint satisfaction, so we explicitly allow it here (following Legendre, Sorace and Smolensky 2006). In tableau (6) output candidate o_{31} violates constraint C_1 twice, but it also positively satisfies C_3 once. There are various ways in which constraints might be formulated so that they assign negative scores on violation and/or positive scores on satisfaction; the details are irrelevant to our convergence proofs and tests.[6] We do note, though, that positive satisfactions seem to be more compatible with HG than with OT (see e.g. Boersma and Escudero in press, Boersma and Pater 2007).

(6) *A more complex illustrative HG evaluation*

i_3	C_1 39.0	C_2 28.0	C_3 14.0	C_4 13.0	C_5 −4.0	
o_{31}	−2		+1	+1	−1	−47.0
o_{32}		−1		−2		−54.0
☞ o_{33}		−1	−1		−2	−34.0
o_{34}	−1		−1			−53.0

Since output candidate o_{31} violates C_1 twice, and the constraint has a weight of 39.0, an amount of −78.0 is contributed to the harmony of the candidate. The candidate does not violate or positively satisfy constraint C_2, hence there is no contribution to the harmony from this constraint. Since candidate o_{31} satisfies both C_3 and C_4 positively, these satisfactions together contribute an amount of $14.0\cdot(+1) + 13.0\cdot(+1) = +27.0$ to the harmony of o_{31}. Candidate o_{31} violates C_5 once, but because the constraint has a negative weight, this "violation" contributes a positive amount of $-4.0\cdot(-1.0) = +4.0$ to the total harmony of o_{31}, which is therefore $-78.0 + 27.0 + 4.0 = -47.0$. The harmonies of the remaining three candidates are computed in analogous ways. Candidate o_{33} ends up with the highest harmony, so it is the winner in the tableau and the actual output of the HG grammar for input i_3.

This tableau provides an example of the fact that in HG, unlike OT, violations of a constraint with lower weight can "gang up" to overcome a violation of a constraint with higher weight: in a comparison of candidate o_{32} to candidate o_{33}, the double violation of C_4 weighs heavier than the single violation of C_3 plus the double "violation" of C_5 (see Legendre, Sorace, and Smolensky 2006, Pater, Bhatt, and Potts 2007, and Tesar 2007 for discussion of the typological consequences of such gang effects).

To illustrate how the learning algorithm works for more complicated instances of constraint violation and satisfaction, let us imagine that a learner is presented with the

[6] Given an infinite candidate set as in Prince and Smolensky 1993/2004, positively satisfied constaints can lead to situations in which no optimum exists because there is no maximum satisfaction score on a constraint (Prince 2007a: fn. 9). To prevent this problem, one has to devise the constraint set and/or the candidate generator with more care than in the violation-only case.

input-output pair (i_3, o_{31}), in other words, the learner is told that the first candidate in (6) is correct. Since the learner now regards o_{33} as the incorrect optimum, the resulting error vector is as in (7).

(7) *Error vector from the constraint scores in (6)*

Correct ~ incorrect	C_1	C_2	C_3	C_4	C_5
(i_3-o_{31}) ~ (i_3-o_{33})	–2	+1	+2	+1	+1

These are the values that will be scaled by the plasticity ε and then added to the constraint weights. This vector illustrates the important point that the degree of change of a constraint's weight in this learning algorithm is sensitive to the degree of difference in satisfaction scores between the correct form and the incorrect optimum. As the correct form satisfies C_3 two times more than the incorrect optimum does, the weight of C_3 will be raised by 2ε, whereas the weight of C_2 will only be raised by ε.

In the version of HG proposed in Legendre, Miyata, and Smolensky 1990 and Smolensky and Legendre 2006, constraints can be both satisfied and violated, and can have both negative and positive weights. It is possible to limit constraints to assigning only negative numbers (violations) or positive numbers (satisfactions), and weightings can be limited to non-negative values (Keller 2000, 2006) or to positive ones (Prince 2002). In our proof, we assume the original, most general, formulation of the theory. In §3.5 we discuss the motivation for banning negative weights, and test learners that impose an automatic positivity restriction on weights (Exponential HG).

3 Convergence of HG-GLA for Nonnoisy Learners

In this section, we first provide a formal definition of the learning algorithm described in §2 (HG-GLA), restricted to learners without evaluation noise. We then provide a convergence proof for such nonnoisy learners, and finally show by computer simulation that the algorithm converges very fast in practice.

3.1 Formalization of the Language Data and the Learner

Our proof operates on the basis of a finite set of language data that we call the dataset. The learning data that are supplied to the learner are sampled from that finite set. Our proof guarantees convergence for any learning simulation that is conducted in this fashion, as are all published OT learning simulations that we are aware of. In section 5, we discuss how an HG learner generalizes from a finite set of learning data to the infinite set of data characterized by a language (and see fn. 12 on directly sampling from the infinite set itself).

The dataset, then, consists of a finite number M of "correct" input-output pairs. Thus, the dataset will have M different input (e.g. underlying) forms i_m (m = 1...M), each of which is coupled with a unique "correct" output form o_m.

We assume that the language user, especially the learner, maintains a finite number K of constraints C_k (k = 1..K) with weights w_k; these weights are used for input-output processing, as in (1) and (6), and can change as a result of learning, as in

(4) and (7). We have five assumptions about the learner's processing, and three about learning.

The first assumption about processing is that for each input i_m of the dataset the learner can compute (or is given) a finite number N_m of output candidates o_{mn} (n = 1.. N_m), as in the tableaux of §2.[7] The second assumption is that the learner can compute (or is given) the extent to which each input-output pair (i_m, o_{mn}) satisfies each constraint C_k (k = 1..K). We call the extent of constraint satisfaction s_{mnk}; for instance, the value of s_{324} in tableau (6) is −2. The third assumption is that the learner can compute a harmony value for each input-output pair (i_m, o_{mn}), as a weighted sum of satisfaction scores:

(8) $$H_{mn} \equiv \sum_{k=1}^{K} w_k s_{mnk}$$

In (6), for instance, the value of H_{32} is −54.0. The fourth assumption is that the learner, when asked to evaluate an input form i_m, can identify the (one or more) *optimal* forms o_{mr}, i.e. the forms that have a maximum harmony among the output candidates:

(9) $$\forall n = 1..N_m : \quad \sum_{k=1}^{K} w_k s_{mrk} \geq \sum_{k=1}^{K} w_k s_{mnk}$$

The fifth assumption about processing is that if there are multiple optimal forms, the learner is able to randomly select one of them as the *winning candidate* of her evaluation.

The objective of the learning procedure is to find constraint strengths w_k such that the optimal outputs for all inputs i_m in the learner's grammar are identical to the correct outputs o_m of the dataset. The first assumption about learning is that for every input-output pair (i_p, o_p) that arrives in the language data, the learner computes her own winning output candidate for the given input i_p. Thus, this winning candidate is o_{pr}, for some r in the range 1..N_p; it has constraint satisfactions s_{prk} and harmony H_{pr}. The second assumption is that the output form o_p given in the language data is among the learner's set of output candidates for i_p, i.e., the learner can identify the given output form as o_{pq}, where $1 \leq q \leq N_p$, and thereby establish the satisfactions s_{pqk} and harmony H_{pq} of this form in her current grammar. The third assumption about learning is that the learner compares this form o_{pq} with her own winning candidate o_{pr}. If it turns out that the two forms are different (and therefore indicate that the learning objective has not been reached yet), the learner labels the situation as an *error*: the form o_{pq} is now considered the *correct output* and o_{pr} the *incorrect optimum*. The learner then takes action by changing the weights of the constraints from w_k to w_k':

(10) $$w_k' = w_k + \varepsilon \cdot \left(s_{pqk} - s_{prk} \right)$$

[7] The finiteness of the candidate set is required for the definition of the margin of separability in §3.2 and D_{max} in §3.3. Under some circumstances, a finite set of possibly optimal candidates might be derived from an infinite candidate set (see Riggle 2004 on "contenders" in OT).

where ε is the plasticity, a positive real number. This update rule is the same as we used in (4), where we took $\varepsilon = 0.4$; the formula is identical to that used by Jäger (2003/to appear) for Maximum Entropy grammars and a notational variant of the formulas used by Soderstrom, Mathis and Smolensky (2006: eqs. 14, 18, 21, 35d) for their connectionist implementation of HG. The update rule implies that the learner raises the weights of all the constraints satisfied better with the correct output o_{pq} than with the incorrect optimum o_{pr}, and lowers the weights satisfied better with the incorrect optimum than with the correct output.

3.2 The Unit Sample Target Grammar

In this section we introduce two concepts that we need for the perceptron convergence proof in §3.3: the *unit sample target grammar*, and one of its properties, the *margin of separation of harmony*.

No learning algorithm can learn a language that its underlying grammar model cannot represent. Thus, the learning algorithm defined in §3.1 will fail to learn any language that cannot be described by a Harmonic Grammar. What we really want to prove, therefore, is that HG-GLA is successful for any set of unique optima that can be generated by a Harmonic Grammar.

By safe assumption, therefore, there exists at least one weighting of the constraints that makes the set of correct forms in the dataset uniquely optimal. Given the nature of the weights (namely, real numbers) and the nature of the interaction of the finite set of K constraints, there probably exist many such weightings, but here we select just one, and call it the *sample target grammar*. The goal of the learner, then, is to arrive either at this sample target grammar or at any other grammar that describes the dataset correctly.

We do not know much about the constraints' weights in the sample target grammar, but we do know that not all weights are zero, because if they were, all candidates would be optimal, and the sample target grammar would exhibit variation in its outputs (unless all inputs have only a single output candidate, but in that case the learner already has a correct grammar from the start). Since not all weights are zero, we can normalize the sample target grammar by dividing every constraint weight (strength) by the square root of the sum of the squares of the constraint weights. Thus, if the sample target grammar has six constraints with weights 17, 5, −3, 14, −5, and 9, the sum of their squares is $17^2+5^2+(-3)^2+14^2+(-5)^2+9^2 = 625$, whose square root is 25. When we divide all constraint weights by 25, we get the weights (0.68, 0.20, −0.12, 0.56, −0.20, 0.36). This grammar generates exactly the same language as it did before the division. We call such a form of the grammar a *unit grammar*: the sum of the squares of its weights is 1. For the unit sample target grammar, we write these weights as u_k (k=1..K).

In the sample grammar, the optimal outputs are unique (they are the correct language forms). Therefore, the harmony (in the sample grammar) of any optimal form is greater than that of any other candidates of the same input. Suppose that for each input i_m the optimal output is o_{mn_m} (n_m is the index of this output in the learner's tableau for i_m). Then

(11) $$\forall m = 1..M, n = 1..N_m, n \neq n_m : \quad \sum_{k=1}^{K} u_k s_{mn_m k} > \sum_{k=1}^{K} u_k s_{mnk}$$

We now assume that for every input a second-best candidate can also be found (it does not have to be unique); it is the winner of the partial tableau that results if the best form is left out. The harmony differences between the second-best forms and the best forms are δ_m (m=1..M); for input i_m, the harmony (in the unit target grammar) of all non-optimal candidates is at least δ_m less than the harmony of the optimal candidate. Since the number M of inputs is finite, there must exist a value δ that is the minimum of all M values δ_m. We therefore know that the harmony of any non-optimal candidate for any input is separated by at least δ from the harmony of the optimal candidate for that same input (in the unit target grammar):

(12) $$\forall m = 1..M, n = 1..N_m, n \neq n_m : \quad \sum_{k=1}^{K} u_k s_{mn_m k} \geq \sum_{k=1}^{K} u_k s_{mnk} + \delta$$

Adapting terminology from the perceptron literature, we call δ the *margin of separation of harmony*; this value is independent of the language data fed to the learner, and it is greater than 0 for any Harmonic Grammar that generates a nonvariable language with a finite number of possible inputs.

3.3 A Perceptron Proof Applied To HG-GLA

The perceptron is a model of linear classification developed by Rosenblatt (1958). Its learning rule is convergent: if two sets of data are linearly separable, the perceptron update rule is guaranteed to find a set of weights that will correctly separate them (Block 1962, Novikoff 1962). Our convergence proof for HG-GLA closely follows the application of the perceptron convergence proof to a case of part-of-speech tagging by Collins (2002), and extends it to the case of HG learning from any initial set of constraint weights, using the grammar and learning model we have just described.[8]

We prove convergence by showing (1) that given any finite set of language data that can be generated by an HG grammar, the learner is guaranteed to make only a finite number of errors, (2) that after having made the last error, the learner has reached a correct grammar, and (3) that the last error will arrive within finite time.

The first goal of our proof, then, is to show that for any given set of language data, we can define an upper bound on the number of errors that the learner will make.

The initial state of the learner (i.e. the state after zero errors) is characterized by a set of initial constraint weights $w_k(0)$. These weights could be all zero, or all 10.0, or weights corresponding to any of the initial rankings proposed in the OT literature (e.g. all markedness constraints high, all faithfulness constraints low, as proposed by Demuth 1995, Levelt 1995, Ohala 1996, Smolensky 1996; see further §6).

[8] In proving convergence for the update rule (10) in a constraint-weighting model, we are preceded by Fischer (2005), who provides a proof of its convergence for Maximum Entropy grammars. Unfortunately, this proof does not immediately generalize to Noisy HG learners, whereas the perceptron proof does (as we show in §4). Within HG, Soderstrom, Mathis and Smolensky (2006) did use the update rule (10), but provided no convergence proof.

As the language data come in, the learner receives a sequence of 'correct' input-output pairs. However, since only the pairs that cause errors will make changes to the weights, we here consider only the pairs that cause an error. It therefore makes sense to define $w_k(t)$ as the weight of constraint C_k after t errors; the set of $w_k(t)$ for all k and all t therefore defines the entire history of the learner's grammar. Let the tth error-causing input-output pair be ($i_{p(t)}$, $o_{p(t)}$), the incorrect winning candidate $o_{p(t)r(t)}$, and the correct output form $o_{p(t)q(t)} = o_{p(t)}$. The update rule for this data pair is then

$$w_k(t) = w_k(t-1) + \varepsilon \cdot \left(s_{p(t)q(t)k} - s_{p(t)r(t)k} \right) \tag{13}$$

We can now start off on some mathematical "tricks" that together allow us to establish an upper bound on the number of errors. First, we take the inner product of all terms in (13) with the weights of the unit sample target grammar:

$$\sum_{k=1}^{K} u_k w_k(t) = \sum_{k=1}^{K} u_k w_k(t-1) + \varepsilon \cdot \left(\sum_{k=1}^{K} u_k s_{p(t)q(t)k} - \sum_{k=1}^{K} u_k s_{p(t)r(t)k} \right) \tag{14}$$

With the help of (12), we can rewrite the last term, so that (14) becomes

$$\sum_{k=1}^{K} u_k w_k(t) \geq \sum_{k=1}^{K} u_k w_k(t-1) + \varepsilon \cdot \delta \tag{15}$$

By induction, starting with $t = 0$, we derive

$$\sum_{k=1}^{K} u_k w_k(t) \geq \sum_{k=1}^{K} u_k w_k(0) + \varepsilon t \delta \tag{16}$$

The first term on the right is the inner product of the weight vector u_k of the unit sample target grammar and the initial weight vector $w_k(0)$ of the learner's grammar. We abbreviate it as $W_0 \cdot U$, so that (16) can be written as

$$\sum_{k=1}^{K} u_k w_k(t) \geq \varepsilon t \delta + W_0 \cdot U \tag{17}$$

According to the Cauchy-Schwarz inequality, the absolute value of the inner product of two vectors is always less than or equal to the product of the norms of the two vectors:

$$\sqrt{\sum_{k=1}^{K} u_k^2} \sqrt{\sum_{k=1}^{K} w_k^2(t)} \geq \left| \sum_{k=1}^{K} u_k w_k(t) \right| \tag{18}$$

Since the first factor on the left in (18) is 1 (since u_k defines a unit grammar, its norm is 1), we can combine (17) and (18) into

$$\sqrt{\sum_{k=1}^{K} w_k^2(t)} \geq \varepsilon t \delta + W_0 \cdot U \tag{19}$$

Equation (19) is valid only for values of t that exceed $-W_0 \cdot U/(\varepsilon\delta)$, which can be a positive time if e.g. the learner's initial grammar has negative weights while the sample target grammar has positive weights. Loosely speaking, (19) says that the overall constraint weights move away from zero with time, i.e. with the number of mistakes t (at least for $t \geq -W_0 \cdot U/(\varepsilon\delta)$). The term on the left in (19) is the norm of w_k. Equation (19) expresses the fact that the norm is bounded from below by a line that rises with t. Equivalently, we can say that the square of the norm is bounded from below by a concave parabola (see Fig. 1):

$$\sum_{k=1}^{K} w_k^2(t) \geq \left(\varepsilon t\delta + W_0 \cdot U\right)^2 \tag{20}$$

at least for $t \geq -W_0 \cdot U/(\varepsilon\delta)$.

Beside a lower bound we can also determine an *upper* bound on this squared norm. We start by writing it in terms of what it was before the latest learning step:

$$\begin{aligned} &\sum_{k=1}^{K} w_k^2(t) = \sum_{k=1}^{K}\left(w_k(t-1) + \varepsilon \cdot \left(s_{p(t)q(t)k} - s_{p(t)r(t)k}\right)\right)^2 = \\ &= \sum_{k=1}^{K} w_k^2(t-1) + 2\varepsilon\sum_{k=1}^{K} w_k(t-1)\left(s_{p(t)q(t)k} - s_{p(t)r(t)k}\right) + \varepsilon^2\sum_{k=1}^{K}\left(s_{p(t)q(t)k} - s_{p(t)r(t)k}\right)^2 \end{aligned} \tag{21}$$

According to (8), the factor after 2ε involves a harmony difference:

$$\sum_{k=1}^{K} w_k(t-1)\left(s_{p(t)q(t)k} - s_{p(t)r(t)k}\right) = H_{p(t)q(t)}(t-1) - H_{p(t)r(t)}(t-1) \tag{22}$$

This is the difference between the harmony of the correct form in the learning data (a non-optimal, or perhaps tied, candidate in the learner's current grammar) and the incorrect optimum (an optimal candidate in the learner's current grammar before the learning step), and must therefore be less than or equal to zero. We can now bound the squared norm in (21) from above by throwing away the 2ε term from (21):

$$\sum_{k=1}^{K} w_k^2(t) \leq \sum_{k=1}^{K} w_k^2(t-1) + \varepsilon^2\sum_{k=1}^{K}\left(s_{p(t)q(t)k} - s_{p(t)r(t)k}\right)^2 \tag{23}$$

We can proceed by noting that the sum in the last term is again bounded from above: it can never be more than the sum of squares of maximum constraint satisfaction differences between the optimal candidates and their nonoptimal counterparts. If the maximum number of constraint violations in the unit grammar for any (potentially incorrectly optimal) input-output pair is V_{max}, and the maximum number of positive constraint satisfactions for any input-output pair is S_{max}, the sum in the last term can never be more than K times the square of the sum of V_{max} and S_{max}. For instance, if no cell in any of the M tableaux contains more than 5 violations or 3 positive satisfactions, the last term in (22) can never be more than $8^2\varepsilon^2K$. Generally, (23) becomes

(24) $$\sum_{k=1}^{K} w_k^2(t) \le \sum_{k=1}^{K} w_k^2(t-1) + \varepsilon^2 K D_{max}^2$$

where D_{max} is defined as $V_{max} + S_{max}$. This value is independent of the language data fed to the learner, and is guaranteed to exist, since the number of output candidates for each input is finite.

By induction on t, starting again at $t = 0$, we get

(25) $$\sum_{k=1}^{K} w_k^2(t) \le \varepsilon^2 K D_{max}^2 t + \sum_{k=1}^{K} w_k^2(0)$$

The last term on the right in (25) is the squared norm of the learner's initial weight vector. We abbreviate it as $|W_0|^2$, so that (25) becomes

(26) $$\sum_{k=1}^{K} w_k^2(t) \le \varepsilon^2 K D_{max}^2 t + |W_0|^2$$

This formula establishes that the squared norm of the learner's weight vector lies below a line that rises with the number of errors t (see Fig. 1). But in (20) we have seen that the squared norm also lies *above* a function that increases quadratically with t. Since a rising parabola cannot stay below a rising line forever with increasing t (see Fig. 1), the two bounding conditions together necessarily indicate that t itself must be limited. More formally, we can combine (20) and (26) to yield

(27) $$\left(\varepsilon\delta t + W_0 \cdot U\right)^2 \le \varepsilon^2 K D_{max}^2 t + |W_0|^2$$

for $t \ge -W_0 \cdot U/(\varepsilon\delta)$. Equation (27) is a quadratic equation in t that can be solved:

(28) $$\varepsilon^2\delta^2 t^2 - \left(\varepsilon^2 K D_{max}^2 - 2\varepsilon\delta W_0 \cdot U\right)t + \left(W_0 \cdot U\right)^2 \le |W_0|^2$$

(29) $$\left(t - \left(\frac{K D_{max}^2}{2\delta^2} - \frac{W_0 \cdot U}{\varepsilon\delta}\right)\right)^2 \le \left(\frac{K D_{max}^2}{2\delta^2} - \frac{W_0 \cdot U}{\varepsilon\delta}\right)^2 + \frac{|W_0|^2 - (W_0 \cdot U)^2}{\varepsilon^2\delta^2}$$

The first thing to establish is that the right-hand side of this equation cannot be negative: if we write $t = 0$ in (18), and take into account that the norm of u_k is one, we see that $|W_0 \cdot U|$ must be less than or equal to $|W_0|$, so that the second term on the right in (29) must be positive or zero. We are therefore allowed to rewrite (29) as

(30) $$\left|t - \left(\frac{K D_{max}^2}{2\delta^2} - \frac{W_0 \cdot U}{\varepsilon\delta}\right)\right| \le \sqrt{\left(\frac{K D_{max}^2}{2\delta^2} - \frac{W_0 \cdot U}{\varepsilon\delta}\right)^2 + \frac{|W_0|^2 (W_0 \cdot U)^2}{\varepsilon^2\delta^2}}$$

Since (30) is valid for all $t \ge -W_0 \cdot U/(\varepsilon\delta)$, it is certainly valid for all $t \ge K D_{max}^2/(2\delta^2) - W_0 \cdot U/(\varepsilon\delta)$. For those values of t, (30) is equivalent to (31):

(31) $$t \leq \frac{KD_{max}^2}{2\delta^2} - \frac{W_0 \cdot U}{\varepsilon\delta} + \sqrt{\left(\frac{KD_{max}^2}{2\delta^2} - \frac{W_0 \cdot U}{\varepsilon\delta}\right)^2 + \frac{|W_0|^2 (W_0 \cdot U)^2}{\varepsilon^2\delta^2}}$$

For smaller values of t, i.e. $t < KD_{max}^2/(2\delta^2) - W_0 \cdot U/(\varepsilon\delta)$, (30) is not equivalent to (31), but (31) holds a fortiori. Therefore, (31) expresses a true upper bound on t.

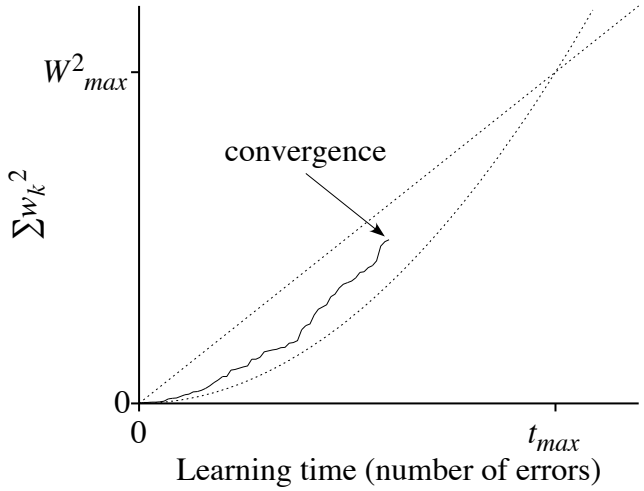

Fig. 1. A possible learning path (in terms of $\Sigma_k w_k^2$) if all weights start at zero. The path stays between the parabola of equation (20) and the line of equation (26). Convergence is reached well before $t_{max} = KD^2_{max}/\delta^2$. The value of W^2_{max} is $\varepsilon^2 K^2 D^4_{max}/\delta^2$.

The interpretation of (31) is that t can never be greater than a value t_{max} (also visible in Fig. 1) that is defined as the right-hand side of (31). This t_{max} is determined completely by properties of the hypothetical unit target grammar (D_{max}, K, U, δ), properties of the initial state ($w_k(0)$), and fixed a priori properties of the learning algorithm (ε); crucially, t_{max} does not depend on any properties of the language data as they are stochastically presented to the learner. This means that given an initial state and a target grammar, we know that the learning algorithm can never make more than a fixed number of t_{max} mistakes. The interpretation is that learning will stop after at most t_{max} mistakes. In other words, the learner will end up in a never-changing grammar after at most t_{max} mistakes.

What will this final stable grammar look like? Well, it is guaranteed to be a correct grammar for the target dataset. For if it were not, the grammar would have to be one that is able to produce an incorrect output for at least one input. That input is certain to arrive at the learner at some point in time (under a common assumption addressed in the next paragraph); when it does, it will cause a mistake and must force learning, which is impossible because learning has stopped. Hence, the final grammar is a correct grammar of the target data set.

The last question is whether the final grammar will ever be reached. After all, t is not the time but the number of mistakes. Between any two consecutive mistakes there may be any number of input-output pairs that cause no mistake, i.e. input-output pairs for which the learner's current grammar produces the same output as that given by the language data. In fact, the likelihood that an input-output pair generates a mistake probably decreases as the learner's grammar approaches one that is appropriate for the

target dataset; as a result, the average expected time between two consecutive mistakes tends to rise as acquisition proceeds. A necessary condition for convergence, therefore, is the same as the one mentioned by Tesar and Smolensky (1998: 246) for the CD convergence proof, namely that informative input-output pairs "are not maliciously withheld from the learner". For the finite dataset under discussion, the continual availability of all forms to the learner is guaranteed if input-output pairs are presented in an infinite sufficiently random sequence. Similarly, any incorrect form that in the learner's grammar ties for harmony with a correct form, should not be maliciously withheld from being chosen as the incorrect winning candidate; the random selection between ties mentioned in §3.1 guarantees this.[9]

Since the number of possible inputs M, the number of output candidates per input N_m, and the number of constraints K are finite, D_{max} in (31), which is the maximum number of constraint violations-minus-satisfactions in at least one correct grammar, will be finite as well. Therefore, t_{max} will have a finite value if a margin of separation δ exists in at least one grammar of the target language; the hypothetical sample target grammar of §3.2 is just such a grammar, again as a result of the finiteness of all the dimensions involved. This rounds up our proof: HG-GLA learners will converge in finite time, regardless of the weights in their initial grammars.

3.4 Computer Simulation: Nonnoisy Learners

It is one thing to show that a learning algorithm converges within a finite time, but quite another thing to show that it converges within some time that would be a realistic characterization of a human learner's youth. Depending on the characteristics of a learning problem, the upper bound on the number of errors expressed by t in (31) can be quite large. We address this issue by showing with computer simulations that the learning algorithm converges in a reasonable amount of time.

To test whether a nonnoisy HG learner converges not just in theory but also in practice, and to test how fast it converges, we generate a random target grammar and a random dataset, as follows. We create the grammar as a set of K constraints (with K randomly drawn from the range 2...20), with weights randomly drawn from a uniform distribution between 2.0 and 18.0. We then define M inputs i_m (M randomly drawn from 1...20) each with N_m output candidates (N_m randomly drawn from 2...20 for each input separately), with each candidate violating each constraint between 0 and D_{max} times (D_{max} randomly drawn from 1...5 for the whole language), as determined by an evenly distributed random choice from among these D_{max}+1 possibilities. That is, we have M tableaux with up to 20 candidates each, each with random integer satisfaction scores from 0 to $-D_{max}$ on each of K constraints. We then use the constraint weights to select the optimal output for each tableau. If an output happens not be unique (which can happen if the best two candidates for that input have the same violation pattern), we discard the whole dataset and start again with new inputs, weights, and violation patterns (and a new K, M, and D_{max}). If necessary, this is repeated until all outputs are unique, so that we know that at least one target grammar exists (it is the grammar we

[9] This condition was not addressed in the EDCD convergence proof by Tesar (1995), and therefore led to misconvergences for that algorithm (fn. 1); as here, the solution for EDCD lies in the random selection between ties (Boersma 2008).

just created). In this way, we create a dataset consisting of *M* "correct" input-output pairs.

Once the target grammar exists, we create a nonnoisy HG learner with the same inputs, output candidates, and violation patterns as the target grammar, but with all constraints weighted at $w_k(0) = 10.0$ (k = 1..K). We feed this learner "correct" input-output pairs of the target language, randomly and evenly selected from among the *M* possible "correct" input-output pairs in the dataset. The learner's grammar evaluates each input with the HG evaluation mechanism, and if the optimal output thus determined is different from the "correct" output given by the target language, the learner takes an HG learning step with a plasticity of $\varepsilon = 1.0$.[10] After every 10 pieces of learning data, we check whether the learner has arrived in a grammar in which all outputs are correct and uniquely optimal. If the learner has reached such a grammar at e.g. the 18th check, we conclude that the learner has converged between the 171st and 180th learning datum.

We perform this procedure not just for a single virtual learner, but for 100,000 virtual learners. They all converge, and do not take much time. Figure 2 shows a histogram of the convergence times.

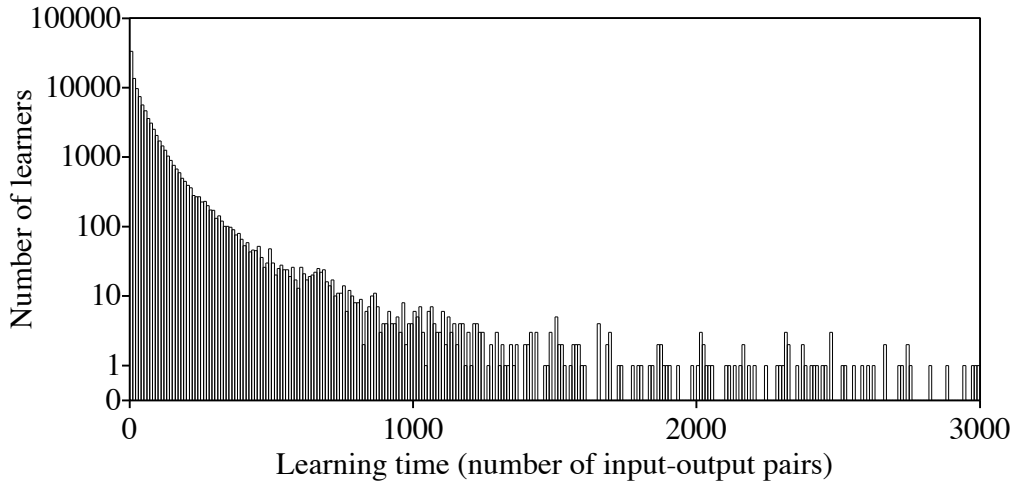

Fig. 2. Histogram of convergence times of 100,000 nonnoisy HG learners. The vertical scale is logarithmic.

The slowest learner requires 24530 pieces of data, but most learners require less than 1000 pieces of data to converge.

HG learners should not only be able to learn from datasets generated by HG grammars, but also from datasets generated by OT grammars, because any finite dataset generated by OT can also be generated by HG. We therefore test another 100,000 nonnoisy HG learners on datasets generated by target OT grammars. We create such a target grammar as a random total ranking of constraints, and compute the *M* optimal outputs in the usual OT way. Again, all 100,000 HG learners converge,

[10] In a computer implementation of a nonnoisy learner, the plasticity has to have an integer value, so as to avoid floating-point rounding errors. This is necessary to allow the learner to go through stages with tied candidates, i.e. stages where multiple candidates within a tableau have exactly equal harmonies. Such situations lead to random variation, and hence to more learning (§3.1).

although it takes them on average slightly more time than they need for the target HG grammars.

We conclude that in practice nonnoisy HG-GLA learners converge quite fast.

3.5 Tests of Learners with a Positivity Restriction: Exponential HG

If HG is to function as an OT-like theory of language typology, then it appears necessary to ban negative weights (Keller 2000, 2006, Prince 2002, Pater, Bhatt, and Potts 2007, Pater, Potts, and Bhatt 2007). To see why, consider the tableau in (32), which takes the violation profiles from (1) and applies a new constraint weighting.

(32) *Harmonic bounding subverted*

i_1	C_1 1.0	C_2 −2.0	
o_{11}	−1		−1.0
o_{12}		−1	+2.0
☞ o_{13}	−1	−2	+3.0

In OT, output candidate o_{13} cannot be optimal under any ranking of the constraints, because its violations are a proper superset of those of one of the other candidates (in fact, of both of them); candidate o_{13} is therefore said to be *harmonically bounded* (Samek-Lodovici and Prince 1999, 2005). Prince (2002) points out that such harmonic bounding relations are preserved in HG if constraints are restricted to have positive weights. Technically, if such a positively-weighted HG permits constraints to assess both positive satisfaction and negative violation scores, we can say that a candidate A harmonically bounds another candidate B if and only if A and B are members of the same candidate set, and the error vector produced by subtracting B's constraint scores from those of A contains at least one positive value, and no negative ones (Becker and Pater 2007).

Thus, if we want to preserve OT-like harmonic bounding in HG, we cannot use negative constraint weights. More generally, if a single constraint can be weighted both negatively and positively, it can both reward and punish the presence of a single structure. This constraint would then behave very differently in HG than in OT. If we do not want this sort of behavior, we have to restrict HG to positive weights.

To determine whether, and with what set of weights a set of input-output pairs can be rendered jointly optimal in positively-weighted HG, one can use the *simplex* algorithm of linear programming, as shown by Pater, Potts and Bhatt (2007); this method functions as the HG equivalent of the recursive version of CD, whose inconsistency detection properties are useful for calculating predicted typologies for OT (see Hayes, Tesar and Zuraw 2003 on typology calculations using recursive CD). However, the simplex algorithm is unlikely to form the basis of a realistic model of human learning; fortunately, the learning algorithm discussed in this paper can be adapted for positively-weighted HG, as we now show.

The HG-GLA learning algorithm can be used with a model of grammar that automatically enforces the positivity restriction directly in the harmony function. In

Exponential HG, which has been available in the Praat program since February 2006, the harmony of a representation is not given by (8), but by (33).

$$H_{mn} \equiv \sum_{k=1}^{K} s_{mnk} e^{w_k} \tag{33}$$

Although the "weights" w_k can have positive and negative values, the weighting of the constraints in the harmony function is e^{w_k}, which is always positive. For the learning algorithm, the update rule is the same as in (10). Hence, the learning algorithm may shift the weight (w_k) of a constraint to a negative value; according to (33), however, such a constraint will still always contribute positively to harmony if it assigns rewards, or negatively if it assigns violations.

Every finite OT-generated dataset should be representable in Exponential HG (Prince and Smolensky 1993 [2004:236]). We therefore test 100,000 learners in the fashion described in §3.4 on datasets generated by nonvarying OT grammars. Some care has to be taken to set the plasticity to a reasonable value: since a value of 1.0 (as in §3.4) would increase the contribution of a constraint by a giant factor of approximately 2.7183 at every learning step, we take a moderate plasticity of 0.1, which leads to a factor of 1.105.[11]

It turns out that all 100,000 learners find a correct weighting. This convergence could be expected on the basis of the observation by Magri (2007) that an equivalent combination of grammar model and learning algorithm (the usual linear HG, with a "Multiplicative GLA") has a convergence proof (based on the "multiplicative perceptron"). The learners are not only correct, but fast as well: Figure 3 shows a histogram of the convergence times. The slowest learner needed 1160 input-output pairs to converge.

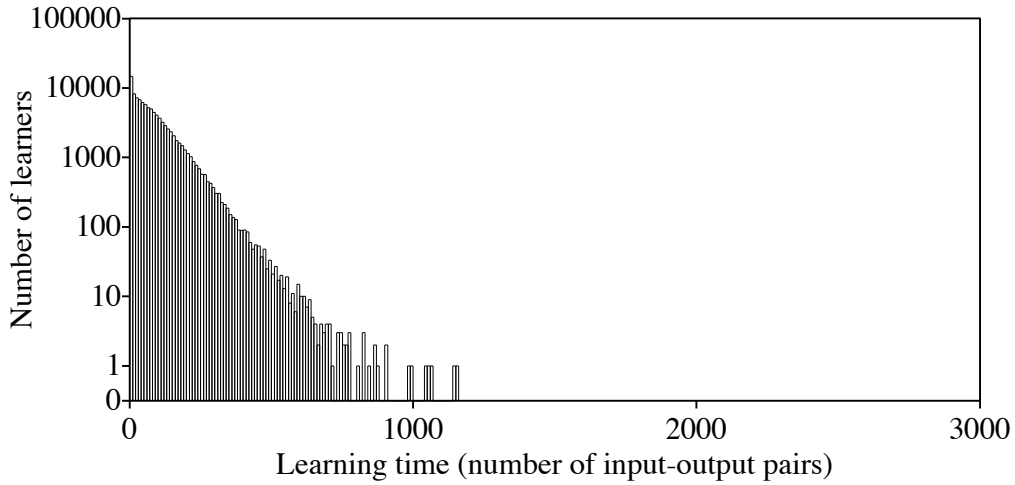

Fig. 3. Convergence times of 100,000 Exponential HG learners on OT-generated datasets.

[11] In a computer implementation, the trick of footnote 10 no longer works. Instead, one can use a tiny evaluation noise, of e.g. 0.001.

4 Convergence of HG-GLA for Noisy Learners

As discussed in the introduction, variation between outputs for the same input can be accounted for by adding noise to constraint weights. In the grammar of such a varying language, as well as in the grammar that the learner maintains during acquisition, the constraint strengths that appear in the evaluations of inputs, such as those written in the top rows of (1) and (2), are not w_k but $w_k + \mathbf{N}_k$, where $\mathbf{N}_k$ is a Gaussian random variable that is temporarily added to each constraint strength at evaluation time, as in (5). This kind of additive noise was introduced to OT by Boersma (1997) and was first combined with HG in Praat in January 2006 (with the correct HG-GLA implemented in April 2007 on the basis of Pater 2008). Jesney (2007), Pater, Bhatt, and Potts (2007), Boersma and Escudero (in press), and Coetzee and Pater (to appear) apply Noisy HG to the modelling of variation. As well as accounting for variation in a target language, noise in the learner's grammar plays a role in modelling gradual acquisition. In combination with a learning algorithm that adjusts the grammar gradually (e.g. OT-GLA or HG-GLA), noisy evaluation produces realistic sigmoid learning curves (see Boersma 1998, Boersma and Levelt 2000 on OT-GLA; also Jäger 2003/to appear for gradual learning in Maximum Entropy grammars).

Section 3 showed that HG-GLA finds correct constraint weights for nonvarying languages if the learner's processing involves no evaluation noise. The present section shows that HG-GLA also finds correct weights for nonvarying languages if the learner's processing does include evaluation noise. We show this first by extending the proof of section 3 to this case. We then provide learning simulations that show that convergence is fast in practice, and compare the HG results with those obtained for OT-GLA with Stochastic OT.

4.1 A Convergence Proof for HG-GLA: Noisy Learners

Instead of by (8), the harmony of an output candidate o_{mn} given an input i_m is now given by

$$H_{mn} \equiv \sum_{k=1}^{K} \left(w_k + \mathbf{N}_k \right) s_{mnk} \tag{34}$$

The lower bound on the sum of the squares of the constraint weights in (20) is still valid. The computation of the upper bound, however, changes, because (22) is no longer true. Instead, it is

$$\sum_{k=1}^{K} \left(w_k(t-1) + \mathbf{N}_k(t-1) \right) \left(s_{p(t)q(t)k} - s_{p(t)r(t)k} \right) = H_{p(t)q(t)}(t-1) - H_{p(t)r(t)}(t-1) \tag{35}$$

Thereby, (23) becomes

(36)

$$\sum_{k=1}^{K} w_k^2(t) \le \sum_{k=1}^{K} w_k^2(t-1) + \varepsilon^2 \sum_{k=1}^{K} \left(s_{p(t)q(t)k} - s_{p(t)r(t)k} \right)^2 - 2\varepsilon \sum_{k=1}^{K} \mathbf{N}_k(t-1) \left(s_{p(t)q(t)k} - s_{p(t)r(t)k} \right)$$

Now suppose that the evaluation noise has a maximum absolute value during acquisition of N_{max}; for instance, if acquisition lasts 100,000 input-output pairs, N_{max} tends to be in the vicinity of $6N$, where N is the standard deviation of the evaluation noise. A lower bound on the value within the last sum in (36) is then $-N_{max}D_{max}$, so that an upper bound of the last sum is $2\varepsilon KN_{max}D_{max}$. Instead of (24) we thus get

$$\sum_{k=1}^{K} w_k^2(t) \le \sum_{k=1}^{K} w_k^2(t-1) + \varepsilon^2 K D_{max}^2 + 2\varepsilon K N_{max} D_{max} \tag{37}$$

In the end, convergence is guaranteed by

$$t \le \frac{\varepsilon K D_{max}^2 + 2K N_{max} D_{max}}{2\varepsilon\delta^2} - \frac{W_0 \cdot U}{\varepsilon\delta} + \sqrt{\left(\frac{\varepsilon K D_{max}^2 + 2K N_{max} D_{max}}{2\varepsilon\delta^2} - \frac{W_0 \cdot U}{\varepsilon\delta}\right)^2 + \frac{|W_0|^2 (W_0 \cdot U)^2}{\varepsilon^2\delta^2}} \tag{38}$$

The value for t_{max} that one can derive from (38) is, other than in (31), no longer independent of the language data stochastically presented to the learner. This is because N_{max} is a property of the learning process rather than a property of the unit target grammar, the initial state, or the learning algorithm. However, the probability that N_{max} exceeds e.g. the non-learning-dependent value of $30N$ during the lifetime of a human learner is vanishingly small (it is in the order of 10^{-190} if the learner hears 100,000,000 utterances). For all imaginable practical purposes, then, t_{max} will exist. Formally speaking, the algorithm converges *almost surely*, i.e. it *converges with probability one*.

4.2 Tests of Convergence for HG-GLA and OT-GLA: Noisy Learners

We now test 100,000 HG-GLA learners on their ability to converge on correct grammars for nonvarying languages, when the learners use evaluation noise in learning. The procedure is the same as that in §3.4, except that the learners incorporate a constraint weight noise with a standard deviation of $N = 2.0$ during the evaluation of every incoming input-output pair. That is, the learner takes the observed input, computes a weighting of its constraints by temporarily adding evaluation noise, and computes the optimal output. If this optimal output is different from the "correct" output that the learner has observed in the given input-output pair, an "error" has occurred and the learner changes some constraint weights. We set the plasticity to a value of 0.1, so that it will take the learner multiple errors to overcome the noise. Again we check the learner's grammar after every 10 learning data (input-output pairs). We judge that the learner has converged to the target *nonvarying* language if we find that the learner's grammar, *if the evaluation noise is set to zero*, produces correct singly optimal outputs for all M possible inputs. Figure 4 shows a histogram of the convergence times.

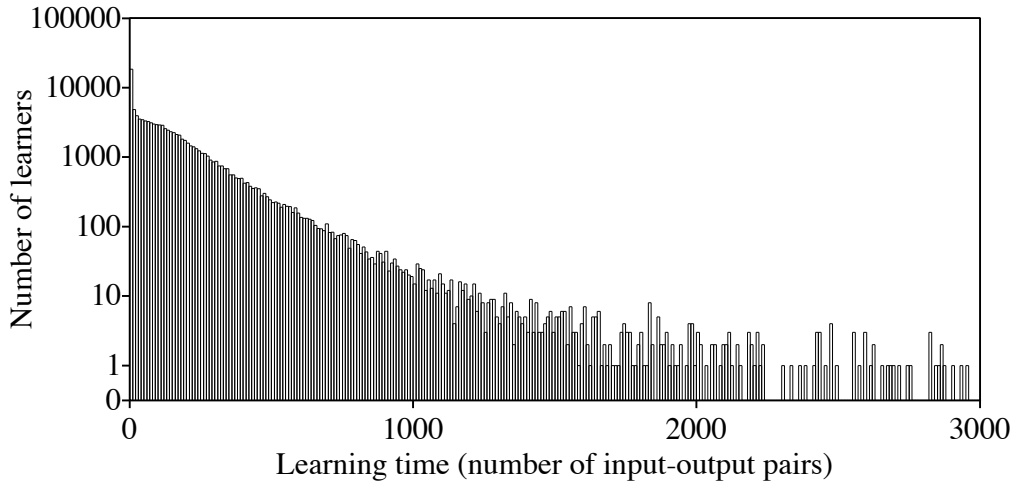

Fig. 4. Histogram of convergence times of 100,000 Noisy HG learners.

All 100,000 Noisy HG learners converge, in times slightly longer (on average) than those of the nonnoisy HG learners of §3.4. The slowest learner needed 82440 input-output pairs.

To get an idea of what a typical learning path looks like, we follow in detail one learner who has $M = 20$ possible inputs. Initially, this learner has all constraints weighted at 10.0 (like all others); for some of the 20 possible inputs, the learner initially already computes a correct output almost 100 percent of the time, if her evaluation noise is 2.0; for some other inputs, she scores close to zero percent correct. For i_{19}, the learner initially scores 12.6 percent correct (this we measure by feeding the learner's grammar 1,000,000 tokens of i_{19}, and dividing the number of correct outputs by 1,000,000). Thus, 12.6 percent is the learner's initial proficiency for i_{19}. We then feed the learner 3,000 learning data randomly drawn from the set of 20 "correct" input-output pairs. After each learning datum, we compute the learner's proficiency in the way just described, i.e. with 1,000,000 tokens of i_{19} and an evaluation noise with a standard deviation of 2.0. Figure 5 shows how the proficiency develops in time. This figure, then, shows this learner's *learning curve* for i_{19}.

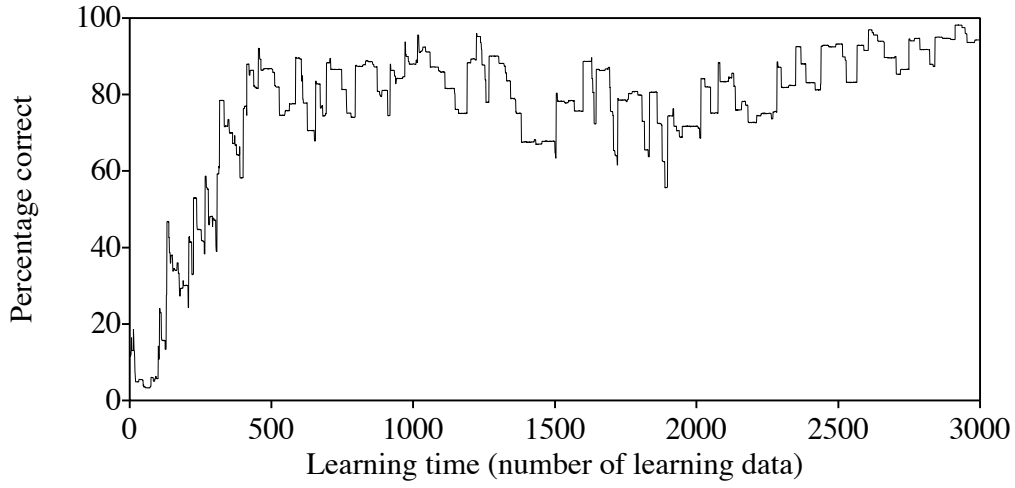

Fig. 5. The learning curve for one learner, for one input.

The learning is jumpy: there are large jumps up when the learner makes an error on i_{19}, and typically small steps up or (more often) down when the learner makes an error on any other input. Some U-shaped learning can also be discerned. In the end, however, the learner moves towards 100 percent correctness on i_{19}, as well as on all the other inputs.

We now examine *Noisy Exponential HG* learners. The harmony function for Noisy Exponential HG combines (33) with (34) and is given in (39).

(39) $$H_{mn} \equiv \sum_{k=1}^{K} s_{mnk} e^{w_k + \mathbf{N}_k}$$

In other words, noise is added to the "weights" w_k at evaluation time. In order to have a moderate variation in the contribution of a constraint to the total harmony of a form, we set the standard deviation of the evaluation noise at 0.2; the plasticity is accordingly set at 0.01. When testing 100,000 Noisy Exponential HG learners on the same OT-generated data as in §3.5, it turns out that all learners are successful. The convergence times are longer on average than for the HG learners of Fig. 4, although the slowest learner requires only 26730 input-output pairs.

Finally, we examine Stochastic OT learners with OT-GLA. OT-GLA differs from HG-GLA in only two ways. First, the grammar model differs in that each time the numerical constraint values are used to evaluate a candidate set, they are converted to a corresponding ranking. Second, the learning procedure differs in the formulation of the update rule. When OT-GLA changes the ranking value of a constraint, it simply moves up or down by an amount ε (if the correct form does worse on the constraint than the incorrect optimum, the constraint moves down; if it does better, the constraint moves up). That is, the HG update rule in (10) is replaced by the OT update rule in (40).

(40) $$w_k' = w_k + \varepsilon \cdot \mathrm{sgn}\left(s_{pqk} - s_{prk}\right)$$

As we noted beneath (7), in the HG update rule the amount of change is proportional to the size of the difference between the satisfaction of the constraint by the correct form and by the incorrect optimum. This difference in the update rules is parallel to the difference in how the grammar models work. Under OT's strict domination, the number of violations of a lower ranked constraint is irrelevant if a candidate is preferred by a higher ranked one, whereas HG permits gang effects, as illustrated in (6).

We thus test Stochastic OT learners with OT-GLA on 100,000 datasets generated by a random total OT ranking: again up to 20 inputs, up to 20 candidates per tableau, up to 20 constraints (all initially ranked at the same height), evaluation noise during learning 2.0, plasticity 0.1. It turns out that only 99.6% of the learners converge. This incomplete convergence of OT-GLA with Stochastic OT is no surprise: it was expected on the basis of a case of nonconvergence discovered by Pater (2008).

One might wonder if the higher convergence rates of Noisy HG with HG-GLA as compared to Stochastic OT with OT-GLA can be attributed solely to a difference in either the grammar models or the learning algorithms. To test this, one can switch

around the learning algorithms between grammar models. It turns out that Stochastic OT has more misconvergence with the HG-GLA update rule than with the OT-GLA update rule, and that with the OT-GLA update rule, Noisy HG learners start to show some misconvergences. Apparently, HG-GLA is the "best" on-line learning algorithm for Noisy HG, and OT-GLA is the "best" learning algorithm for Stochastic OT (until a fully convergent alternative is found).

The fact that HG-GLA continues to converge when noise is added to the learner's grammar model allows it to combine the strengths of OT-GLA (representation of variable languages, learning under noisy data, gradual learning) with the strength of EDCD (guaranteed convergence on nonvarying languages). Further research will be required to better understand the exact nature of the differences between the varying languages produced by Stochastic OT, by Noisy HG, and by the Maximum Entropy approach to stochastic HG (see fns. 5, 8). In addition, further work is required to extend our convergence proof to the case of learning varying languages (as Fischer 2005 did for Maximum Entropy learning). However, in preliminary tests of probability matching that we have performed, we found no noticeable differences between HG-GLA, OT-GLA, and Jäger's (2003/to appear) Maximum Entropy learners (see also Coetzee and Pater to appear).

5 Generalizing from the Finite Dataset

The proofs of §3.3 and §4.1 assume a finite dataset from which the input-output pairs supplied to the learner are randomly drawn. This describes exactly the situation in many, if not all, published OT/HG learning simulations, and the proof is therefore of considerable value as an underpinning of future work in this area. From the perspective of learning theory, however, this assumption raises an important issue.[12] Convergence on a finite dataset may be considered unremarkable, in that one could equally well create a learner that simply memorizes the correct forms. To counter this worry, this section discusses how HG learners handle forms they never encountered before.

Speakers create new sentences all of the time. They routinely concoct underlying forms that have never appeared as an "input" in any input-output pair that they have received during their acquisition period. Learners that simply memorize correct output forms will be at a loss when having to construct a new one. Unlike such learners, OT and HG learners will always be able to compute an optimal output on the basis of their constraint ranking or weighting. Since this ranking or weighting is based on their past experience with a finite dataset, we can say that when dealing with new inputs, these learners *generalize* from a finite dataset to an infinite language. The present section discusses four basic kinds of generalization, showing that in three of these

[12] A more subtle issue is that proofs of convergence following Gold (1967), including the CD convergence proof in Tesar & Smolensky (2000), assume that the learning data are directly sampled from the infinite set of possible forms defined by the target language. Our adaptation of the perceptron proof assumes instead that the learner's data come from a finite dataset, which is itself a sample from the target language. This assumption is made in order to facilitate the definition of a margin of separation (§3.2). We leave the weakening of this assumption for further research.

cases HG learners act in the same way as OT learners, whereas in the one remaining case the two kinds of learners diverge.

Our simplest example of generalization concerns a case in which a new input yields a candidate set with the same violation profiles as that of an input that has occurred in the learning data. Such cases typically occur if violation counts do not depend on the length of the input string. An example appears in (41), which has the constraints ONSET 'assign a violation mark to every V that is not preceded by C' (V = vowel, C = consonant) and DEP 'assign a violation mark to every segment in the output that lacks a correspondent in the input'. In the tableaux, the subscripts depict the correspondence between inputs and outputs, and "." depicts a syllable boundary.

(41) *Generalizing with a single violation profile*

/$V_1C_2V_3$/	ONSET 2.0	DEP 1.0
[.V_1.C_2V_3.]	–1	
√ ☞ [.CV_1.C_2V_3.]		–1

/$V_1C_2V_3C_4V_5$/	ONSET 2.0	DEP 1.0
[.V_1.C_2V_3.C_4V_5.]	–1	
☞ [.CV_1.C_2V_3.C_4V_5.]		–1

On the left-hand side of (41) we see the direct result of learning. During its acquisition period, the learner in (41) has been confronted with the "correct" input-output pair /$V_1C_2V_3$/ → [.CV_1.C_2V_3.] ("√"), and as a result the learner's GLA has moved the weight of ONSET above that of DEP, so that the correct output becomes optimal in the learner's grammar ("☞"). On the right-hand side of (41) we see the result of generalization. Although the input differs from the one seen in learning, the violation profiles of the candidates are identical. Even if the learner has never experienced the input /$V_1C_2V_3C_4V_5$/ before, its grammar will be able to compute an optimal output ("☞"), again prepending a consonant. This is equally true for any other input string that has only one vowel that lacks a preceding consonant. The generalization that the HG learner makes in (41) happens in an identical way for OT learners, who learn the ranking ONSET >> DEP and generalize this onset preference to new inputs.

Our second example of generalization, illustrated in (42), concerns a case in which the new input produces a candidate set whose violation profiles are equivalent, but not identical, to those for an input in the learning data. In this example, violations of two conflicting constraints trade off one-to-one: each violation of ONSET can be avoided by exactly one violation of DEP. As the length of the input string increases, the violations of the constraints tend to grow proportionally to one another.

(42) *Generalizing by scaling the whole violation profile*

/V_1V_2/	ONSET 2.0	DEP 1.0
[.V_1.V_2.]	–2	
√ ☞ [.CV_1.CV_2.]		–2

/$V_1V_2V_3$/	ONSET 2.0	DEP 1.0
[.V_1.V_2.V_3.]	–3	
☞ [.CV_1.CV_2.CV_3.]		–3

If the learner has been confronted (during acquisition) with the pair /V_1V_2/ → [.CV_1.CV_2.], the learner's HG-GLA will have moved ONSET over DEP, as in the

tableau on the left. When the learner is subsequently confronted (after acquisition) with the new input /$V_1V_2V_3$/, the learner will again insert a number of onsets, as in the tableau on the right, because the violation profile is just a scaled-up version of the one on the left. This type of straightforward generalization is also made by OT learners, who learn that ONSET outranks DEP and generalize this onset preference to new inputs. The equivalence between OT and HG under this sort of one-to-one trade-off between constraint violations is discussed by Prince (2002), Pater, Bhatt, and Potts (2007), and Tesar (2007).

Our remaining examples are cases where a single violation of one constraint trades off against an unbounded number of violations of a competing constraint, since the number of violations of the second constraint, but not the first, depends on the length of the string. In tableau (43), the constraint NOCODA 'assign a violation mark to every syllable with a final C' competes with MAX 'assign a violation mark to every segment in the input that has no correspondent in the output'. As we add consonants to the end of a /CVC/ input string, the single violation of NOCODA in the faithful candidate remains constant, while the number of MAX violations needed to avoid the NOCODA violation increases (here we simplify away from the influence of possible constraints like NOCOMPLEXCODA).

(43) *Generalizing by worsening the worst*

/$C_1V_2C_3$/	MAX 2.0	NOCODA 1.0
√ ☞ [.$C_1V_2C_3$.]		−1
[.C_1V_2.]	−1	

/$C_1V_2C_3C_4$/	MAX 2.0	NOCODA 1.0
☞ [.$C_1V_2C_3C_4$.]		−1
[.C_1V_2.]	−2	

If the learner has been confronted with the pair /$C_1V_2C_3$/ → [.$C_1V_2C_3$.], the learner will have moved MAX over NOCODA (on the left). When the learner subsequently has to produce the new input /$C_1V_2C_3C_4$/ (on the right), the learner will realize a faithful output a fortiori, because deleting two underlying segments is even worse than deleting one. Again, this type of generalization is also made by OT learners, who learn that MAX outranks NOCODA and generalize this coda preference to new inputs.

This kind of unbounded trade-off can yield differences between the ways in which HG and OT learners generalize, as we show in our fourth case. The tableaux in (44) illustrate.

(44) *Generalizing by worsening the best*

/$C_1V_2C_3$/	NOCODA 1.5	MAX 1.0
[.$C_1V_2C_3$.]	−1	
√ ☞ [.C_1V_2.]		−1

/$C_1V_2C_3C_4$/	NOCODA 1.5	MAX 1.0
HG: ☞ [.$C_1V_2C_3C_4$.]	−1	
OT: ☞ [.C_1V_2.]		−2

If the learner has been confronted (during acquisition) with the pair /$C_1V_2C_3$/ → [.C_1V_2.], the learner's HG-GLA will have moved NOCODA over MAX, as in the tableau on the left. When the learner is subsequently confronted (after acquisition)

with the new input /$C_1V_2C_3C_4$/, the HG learner may now (depending on the precise weights) no longer delete the whole coda, as in the tableau on the right. OT learners, on the other hand, learn that NoCoda outranks Max and generalize this coda aversion to new inputs, invariably choosing the truncated candidate. Thus, HG and OT make different predictions for generalization.

Testing these predictions can be difficult insofar as the contents of the constraint set can vary. The example in (44) is based on a case mentioned by Prince and Smolensky (1997: 1606), who take the prediction of OT to be correct. Pater, Bhatt and Potts (2007: 19) note that the divergence between the theories depends on a definition of NoCoda that assigns a single violation to a string of post-vocalic consonants. If the constraint is defined so that it assigns a penalty to each of the consonants (e.g. 'assign a violation mark to every consonant that is not in onset position'), violations of NoCoda and Max will trade off one-to-one, and the HG and OT learners will generalize in an identical fashion. Pater, Bhatt and Potts (2007) also discuss an example from Japanese loanword phonology in which HG and OT make different predictions, and in which the facts favor HG. Clearly, differences in the typological predictions of the two theories, and in their predictions for generalizations that human learners make, are deserving of further research.

6 The Severity of the Subset Problem

The last section discussed how HG and OT learners generalize from the finite set of target language data that they learn from. The *subset problem* for OT and HG acquisition (Smolensky 1996) can be characterized in terms of such generalization. In this section, we illustrate the subset problem by extending the case of Onset – Dep interaction we have been discussing, and show how differences in the learning algorithm (EDCD *vs*. GLA) and the grammar module (OT *vs*. HG) can lead to differences in how the learner generalizes.

Imagine a language in which all syllables start with a consonant, and in which any vowel-initial underlying forms are repaired by prepending a consonant in the output. Imagine further that a learner happens to have the ranking Dep >> Onset at some point during acquisition, as in (45).

(45) *An informative input-output pair for an onset-requiring language*

/V_1/	Dep	Onset
☞ [.V_1.]		–1
√ [.CV_1.]	–1	

The learner in (45) incorrectly turns an underlying V-initial form into a surface V-initial form ("☞"). Fortunately, all learning algorithms that rely on competition between candidates can solve this problem. When the input-output pair /V_1/ → [.CV_1.], marked by "√" in (45), arrives at an EDCD learner, this learner will immediately demote Dep below Onset; when it arrives at an OT-GLA or HG-GLA learner, such a learner will demote Dep a bit and raise Onset a bit. Ultimately, given the presence of (45) in the dataset (and no contradictory data), all three learners will

converge on a grammar in which ONSET is placed above DEP, rendering /V_1/ → [.CV_1.] correctly optimal.

However, it is sometimes the case that the learning data include no informative pairs like /V_1/ → [.CV_1.]. Such informative pairs are only available if a language has the relevant morphophonemic alternations and the learner can construct underlying representations from such alternations (a process we abstract from; see Apoussidou 2006:ch.6 for a proposal that utilizes the GLA). If a certain language without vowel-initial surface forms (outputs) does not have such alternations, then it is quite likely that the lexicon of this language does not contain any vowel-initial underlying forms (inputs) (see Prince and Smolensky 1993 [2004: 225–231] on *lexicon optimization*). In datasets drawn from such languages, input-output pairs like /V_1/ → [.CV_1.] will never occur; learners like those in (45), with the ranking DEP >> ONSET, will then only receive input-output pairs of the type /C_1V_2/ → [.C_1V_2.], and such data provide no evidence that the ranking should be changed. The learner may thus end up with the final ranking DEP >> ONSET. This is problematic, because such a learner would generalize to new data differently from a human learner: confronted with the new input /V_1/, the learner would map it to the faithful output [.V_1.], while a human would likely map it to [.CV_1.], with the obligatory onset. Evidence that humans indeed map new inputs to outputs that honor the language's output restrictions can be gathered from loanword adaptation, second language acquisition, and laboratory experiments (Davidson, Jusczyk and Smolensky 2004).[13]

As a solution to this subset problem, Smolensky (1996) and Tesar and Smolensky (2000: §5.1) propose that in the learner's initial ranking, Markedness constraints like ONSET are placed above Faithfulness constraints like DEP. With this initial "M >> F" ranking, the learner of an onset-requiring language would not need evidence of an unfaithful mapping like that in (45) to place ONSET over DEP. But, as Hayes (2004) and Prince and Tesar (2004) show, an initial M >> F ranking is insufficient to solve all restrictiveness problems. To stay with the present type of example, one can imagine a language that allows initial onsetless syllables ([.V.CV.]) but no hiatus (*[.CV.V.]). Such a language can be described by assuming two separate DEP constraints: one that penalizes an output consonant in word-initial position that lacks an input correspondent (DEP-INITIAL), and one that penalizes such a consonant in word-medial position (DEP-MEDIAL). Given an initial general M >> F ranking, namely { ONSET } >> { DEP-INITIAL, DEP-MEDIAL }, and learning data with word-initial onsetless syllables (e.g. /$V_1C_2V_3$/ → [.V_1.C_2V_3.]), an EDCD learner would correctly demote ONSET below DEP-INITIAL. However, this automatically also demotes ONSET below DEP-MEDIAL, so that the learner ends up with the ranking { DEP-INITIAL, DEP-MEDIAL } >> { ONSET }. The problem is that this ranking would also allow hiatus (e.g. /$C_1V_2V_3$/ → [.C_1V_2.V_3.]), which does not occur in the language. To deal with this sort of problem, Hayes (2004) and Prince and Tesar

[13] The subset problem could also be labelled as one of misconvergence rather than misgeneralization (Tesar and Smolensky 2000: 100). For example, the input-output pair /V_1/ → [.CV_1.] could be assumed to belong to the target language, even if it never shows up in an actual dataset. The learner can be said to fail to converge on that target language because some kinds of informative input-output pairs are systematically withheld.

(2004) propose versions of recursive CD that maintain biases for low faithfulness thoughout learning. These algorithms are considerably more elaborate than EDCD.

Boersma and Levelt (2003) point out that such subset problems are less severe for OT-GLA than for EDCD, due to the fact that OT-GLA both promotes and demotes constraints. For example, if we give ONSET an initial value of 10, and DEP-INITIAL and DEP-MEDIAL initial values of 0, then repeatedly exposing OT-GLA to the /$V_1C_2V_3$/ → [.V_1.C_2V_3.] mapping will produce a final state in which DEP-INITIAL = 6, ONSET = 4 and DEP-MEDIAL = 0 (if the plasticity is 1). Because DEP-INITIAL is promoted selectively, ONSET remains above DEP-MEDIAL, and unlike the final grammar produced by EDCD, this final grammar rules out medial onsetless syllables (e.g. it maps /$C_1V_2V_3$/ to [.C_1V_2.CV_3.]). HG-GLA yields exactly the same result as OT-GLA here. For this case at least, these algorithms yield grammars that produce only the observed surface forms, even for types of input forms not encountered before.

Jesney and Tessier (2007a) point out that if faithfulness constraints can be in a specific-to-general relation, HG-GLA tends to yield an even more restrictive final grammar than OT-GLA. As an example, we can use ONSET with the specific DEP-INITIAL and the general DEP, with initial M >> F weights (ONSET = 10, DEP = 0, DEP-INITIAL = 0). If the input-output pair /$V_1C_2V_3$/ → [.V_1.C_2V_3.] is then given to an OT-GLA or HG-GLA learner, ONSET will be demoted by 1 and both faithfulness constraints will be promoted by 1. The difference in outcomes between OT-GLA and HG-GLA comes from the fact that HG-GLA will stop making errors when the *sum* of the weights of DEP and DEP-INITIAL exceeds that of ONSET. Given the assumptions about noise and plasticity from the last paragraph, the HG-GLA final state grammar will have ONSET = 6, DEP = 4, and DEP-INITIAL = 4. The correct /$V_1C_2V_3$/ → [.V_1.C_2V_3.] is optimal because insertion of a word-initial consonant violates both faithfulness constraints, and is with this weighting worse than an ONSET violation. Because ONSET remains above DEP, however, medial onsetless syllables are ruled out. OT-GLA, on the other hand, would end up promoting both faithfulness constraints above ONSET, thus yielding a grammar that accepts medial onsetless syllables. EDCD would also produce this less restrictive outcome, as would the biased version of CD in Prince and Tesar 2004 (cf. Hayes 2004, Tessier 2006).

In sum, the structure of both the learning algorithm and the grammar module in HG-GLA can lead to more restrictive learning outcomes than either EDCD or OT-GLA, and even a biased version of CD. This provides reason to be optimistic that the overall severity of subset problems is reduced in this approach to learning, and that complexities analogous to those introduced in the revisions to CD by Hayes (2004) and Prince and Tesar (2004) can be avoided.[14]

[14] Beside providing initial M > F weights, there could be other ways to bias HG-GLA toward final M > F states. Jesney and Tessier (2007ab) provide examples in which reducing the plasticity of faithfulness constraints relative to markedness constraints helps maintain restrictiveness. Another approach may be to let the M constraints drift up and/or the F constraints drift down, or to have the weights of the M and F constraints *decay* to different extents.

7 Learning with Imperfect Knowledge of Right and Wrong

We now turn to a set of learning problems for which there is no convergence proof for either CD or HG-GLA. These are cases in which the learner is not supplied with full knowledge of the structure of the correct forms, that is, when there is *hidden structure*. We provide an initial examination of the convergence properties of HG learners by testing them on a set of hidden structure problems that have already been investigated for OT learners.

These learning simulations build on work by Tesar (1997) and Tesar and Smolensky (2000), who discuss a case where human learners do not have direct access to the full phonological structure of output forms. In the case under discussion (simplified metrical phonology), the overt language data provides learners with a sequence of syllables and with information on which of these syllables have primary, secondary, or no stress, but it does not provide any direct information on where the foot boundaries are (since these are inaudible). Tesar and Smolensky propose that learners use a general procedure called *robust interpretive parsing* (RIP) to infer such hidden grammatical structure from the overt data: in the metrical case, learners use their current constraint ranking to determine the optimal foot structure that is consistent with the overt stress pattern that they heard. The learner then regards this inferred (though possibily non-adultlike) full foot structure as the "correct" form needed in an error-driven learning procedure (which for Tesar and Smolensky is EDCD); the learner also computes (from the inferred underlying form) its own optimal output (foot structure) in production (as usual; see e.g. tableau (2)). When the learner makes an "error", i.e. when the inferred foot structure is different from the learner's own optimal foot structure, the grammar is adjusted on the basis of a comparison of the incorrect optimum with the correct form (as usual).

Using this approach to the learning of hidden structure, Tesar and Smolensky tested EDCD on datasets from 124 different nonvarying OT target languages that used a set of 12 metrical constraints. Their simulations were replicated by Boersma (2003), who also tested Stochastic OT with OT-GLA on the same set of languages. To compare the HG learners with the OT learners, we replicate both of these earlier sets of simulations, and also test noisy and nonnoisy learners that operate with HG-GLA within the normal HG and within Exponential HG. Here we only report the results; for further details of the simulations, see Tesar and Smolensky (2000) and Boersma (2003). Tesar and Smolensky ran their simulations with various assumptions about the initial ranking; our simulations always have the constraints start out at an equal ranking or weighting (a value of 10.0). The nonnoisy learners have an evaluation noise of zero and a plasticity of 1.0; the noisy learners have an evaluation noise of 2.0 and a plasticity of 0.1 (for the Exponential HG learners, all these values are divided by 10, as in §4.2).

We test each combination of grammar type and learning algorithm ten times on each of the 124 languages, providing each of the 1240 learners with a maximum of 1 million overt forms randomly drawn from the 62 possible overt forms of the language. As in §4.2, we judge a learner successful once its grammar, *if the evaluation noise is set to zero*, renders uniquely optimal all of the correct forms of the target language. The results are shown in (46).

(46) *Results of learning simulations with hidden metrical structure*

Learner grammar	Learning algorithm	Performance of nonnoisy learners	Performance of noisy learners
OT	EDCD	46.94%	–
OT	OT-GLA	55.40%	58.95%
HG	HG-GLA	82.02%	88.63%
Exponential HG	HG-GLA	70.89%	88.95%

The relative success rates of EDCD and OT-GLA replicate the findings of Tesar and Smolensky (2000) and Boersma (2003).[15] HG-GLA learners turn out to have higher success rates than OT learners.

For every combination of grammar type and learning algorithm, we see that noisy learners have a higher probability of convergence than non-noisy learners. Apparently, evaluation noise not only allows the grammar to model variable languages, but it can also help to solve difficult learning problems. This outcome may be related to the fact that noise can help optimization algorithms escape local optima (cf. on *simulated annealing* in neural networks and linguistics: Ackley, Hinton and Sejnowski 1985, Hinton and Sejnowski 1986, Smolensky 1986, Legendre, Sorace and Smolensky 2006, Biró 2006). The noisy learners also have more consistent outcomes than the nonnoisy ones: for 108 languages, all 10 Noisy HG learners converge, i.e. these languages can be considered learnable; for 11 languages all 10 learners fail, i.e. these languages are unlearnable; and for only 5 languages, the number of converging learners is between 1 and 9, i.e. these languages are only sometimes learned (and therefore perhaps historically unstable). For nonnoisy HG learners, the number of languages falling into each category is 68, 23, and 33: most of the non-fully learnable languages are still learned by some. The relative consistency of noisy learning allows a better diagnostic of the difficulty of particular languages, which may be useful in amending the grammatical system or the learning algorithm, or in using this difficulty to explain typological gaps (see Boersma 2003).

We emphasize that we have provided only an initial investigation of the hidden structure problem in HG. Even for just this instance, further research is required to understand the sources of the relative success rates of the learners in (46), and the nature of the problems posed by these metrical systems. Nonetheless, for the case of error-driven learning of hidden structure problems, it seems that the "symmetric" OT-GLA and HG-GLA algorithms, which both promote and demote constraints, have an advantage over demotion-only EDCD (also Apoussidou 2007: chs. 4, 5), and that a learner with an HG grammar has an advantage over one using OT ranked constraints. In research subsequent to Tesar and Smolensky (2000), Tesar (2000) proposes the Inconsistency Detection Learner, an algorithm for the learning of hidden structure,

[15] The percentage correct reported here for EDCD is lower than that in Tesar and Smolensky (2000) and Boersma (2003), because that (64 percent correct) was based on learners with constraint strata with "pooling ties". When asked to turn their final strata into totally ranked hierarchies, the performance drops to 32 percent. The percentage reported here is based on the corrected EDCD by Boersma (2008), which uses "permuting ties", making sure that at every stage during acquisition the learner can correctly convert its stratified ranking to a totally ranked hierarchy. See also fn. 7.

which is guaranteed to converge, but at the cost of having to maintain multiple simultaneous grammar hypotheses; also, as Tesar notes (p. 37), it cannot learn variable languages. The overall success of the noisy HG learners on the test cases examined here provides reason to be optimistic that this approach to learning can deal with both hidden structure and variable languages, using a simple learning algorithm that maintains a single grammar hypothesis at a time.

8 Conclusion

This paper provides a set of basic learning results for Harmonic Grammar. We have shown that a simple on-line gradual learning algorithm, HG-GLA, is guaranteed to converge on a correct set of weights for any nonvarying set of HG-generated language data, if it is supplied with full knowledge of the structure of the language data. We have shown that this result holds for both nonnoisy and noisy learners. Like Stochastic OT, Noisy HG therefore provides a grammar model that can represent variation, is robust against mistakes in the learning data, and exhibits gradual learning curves. In addition to these core results, we have discussed initial investigations into the subset problem and the hidden structure problem in HG learning, which provide some indications that there may be advantages for HG over OT in these domains. These results provide a foundation for further research on the formal properties of this approach to learning, as well as research on the modelling of human language acquisition. The further development of HG learning theory has a large body of extant research to draw upon, since HG's linear model is widely used in neural-modelling and statistical approaches to learning. Our results draw on that strength, especially in the adaptation of the perceptron convergence proof.

These results add to a growing body of evidence that HG should be taken seriously as a model of generative grammar. Although Prince and Smolensky (1993 [2004: 232–233]) state that weighted constraints would produce implausible typological results, there has been very little explicit comparison of the power of ranked and weighted constraints (see Legendre, Sorace and Smolensky 2006, Pater, Bhatt, and Potts 2007, Pater, Potts, and Bhatt 2007, Tesar 2007, and Prince 2007b for initial results). HG also seems to have some distinct advantages for linguistic analysis. Pater, Bhatt, and Potts (2007) argue that it provides a more restrictive approach to cumulative constraint interaction than Smolensky's (2006) OT with Local Constraint Conjunction, and Boersma and Escudero (in press) show that HG does better than OT with positive constraint satisfactions.

Regarding these learning results and those of the connectionist-symbolic framework developed in Smolensky and Legendre 2006, one can say that Harmonic Grammar is an area where neural modelling has started to meet generative grammar. Should HG prove successful as a model of generative grammar, this may contribute to meeting Smolensky and Legendre's goal of elaborating a fully integrated cognitive architecture, with an explicit mathematical characterization of the connection between neural processing and higher-level cognition, including linguistic grammar.

References

Ackley, David H., Geoffrey E. Hinton, and Terrence J. Sejnowski. 1985. A learning algorithm for Boltzmann machines. *Cognitive Science* 9:147–169.

Anderson, James, and Edward Rosenfeld (eds.). 1989. *Neurocomputing: foundations of research*. Cambridge, MA: MIT Press.

Anttila, Arto. 1997. Deriving variation from grammar. In *Variation, change and phonological theory*, ed. by Frans Hinskens, Roeland van Hout, and Leo Wetzels, 35–68. Amsterdam: John Benjamins.

Anttila, Arto. 2007. Variation and optionality. In *The Cambridge handbook of phonology*, ed. by Paul de Lacy, 519–536. Cambridge: Cambridge University Press.

Apoussidou, Diana. 2007. The learnability of metrical phonology. Doctoral dissertation, University of Amsterdam.

Becker, Michael, and Joe Pater. 2007. OT-Help user manual. In *University of Massachusetts Occasional Papers in Linguistics 36: Papers in theoretical and computational phonology*, ed. by Michael Becker, 1–12. Amherst, MA: GLSA Publications. [ROA-928]

Biró, Tamás. 2006. *Finding the right words: implementing Optimality Theory with simulated annealing*. Doctoral dissertation, Groningen University.

Block, H. D. 1962. The perceptron: A model for brain functioning. *Reviews of Modern Physics* 34:123–135. [Reprinted in Anderson and Rosenfeld 1989]

Boersma, Paul. 1997. How we learn variation, optionality, and probability. *Proceedings of the Institute of Phonetic Sciences* 21:43–58. University of Amsterdam.

Boersma, Paul. 1998. *Functional phonology: formalizing the interactions between articulatory and perceptual drives*. Doctoral dissertation, University of Amsterdam.

Boersma, Paul. 2003. Review of Tesar & Smolensky (2000): *Learnability in Optimality Theory*. *Phonology* 20:436–446.

Boersma, Paul. 2008. Some correct error-driven versions of the Constraint Demotion algorithm. Ms. University of Amsterdam. [ROA-953]

Boersma, Paul, Joost Dekkers, and Jeroen van de Weijer. 2000. An introduction to Optimality Theory: Phonology, syntax, and acquisition. In *Optimality Theory: phonology, syntax, and acquisition*, ed. by Joost Dekkers, Frank van der Leeuw, and Jeroen van de Weijer, 1–44. Oxford: Oxford University Press.

Boersma, Paul, and Paola Escudero. In press. Learning to perceive a smaller L2 vowel inventory: an Optimality Theory account. In *Contrast in phonology: theory, perception, acquisition*, ed. by Peter Avery, Elan Dresher, and Keren Rice, 271–301. Berlin: Mouton de Gruyter.

Boersma, Paul, and Bruce Hayes. 2001. Empirical tests of the Gradual Learning Algorithm. *Linguistic Inquiry* 32:45–86.

Boersma, Paul, and Clara Levelt. 2000. Gradual constraint-ranking learning algorithm predicts acquisition order. In *Proceedings of Child Language Research Forum 30*, ed. by Eve V. Clark, 229–237. Stanford: CSLI Publications.

Boersma, Paul, and Clara Levelt. 2003. Optimality Theory and phonological acquisition. *Annual Review of Language Acquisition* 3:1–50.

Boersma, Paul, and Joe Pater. 2007. Constructing constraints from language data: the case of Canadian English raising. Paper presented at North East Linguistic Society (NELS) 38, Ottawa. [Available at http://people.umass.edu/pater/boersma-pater-nels.pdf]

Coetzee, Andries, and Joe Pater. To appear. The place of variation in phonological theory. In *The handbook of phonological theory* (2nd ed.), ed. by John Goldsmith, Jason Riggle, and Alan Yu. Malden, MA: Blackwell. [ROA-946]

Collins, Michael. 2002. Discriminative training methods for Hidden Markov Models: theory and experiments with perceptron algorithms. *2002 Conference on Empirical Methods in Natural Language Processing*. [Available at http://people.csail.mit.edu/mcollins/papers/tagperc.pdf]

Davidson, Lisa, Paul Smolensky, and Peter Jusczyk. 2004. The initial and final states: theoretical implications and experimental explorations of Richness of the Base. In *Fixing priorities: constraints in phonological acquisition*, ed. by René Kager, Joe Pater, and Wim Zonneveld, 321–368. Cambridge: Cambridge University Press.

Demuth, Katherine. 1995. Markedness and the development of prosodic structure. In *Proceedings of the North-East Linguistics Society (NELS) 25*, ed. by Jill Beckman. Amherst, MA: GLSA Publications. [ROA-50]

Fischer, Markus. 2005. A Robbins-Monro type learning algorithm for a maximum-entropy maximizing version of stochastic Optimality Theory. Master's thesis, Humboldt University, Berlin. [ROA-767]
Fodor, Janet Dean. 1998. Unambiguous triggers. *Linguistic Inquiry* 29:1–36.
Gibson, Edward, and Kenneth Wexler. 1994. Triggers. *Linguistic Inquiry* 25:407–454.
Gold, Mark. 1967. Language identification in the limit. *Information and Control* 10:447–474.
Goldrick, Matt, and Robert Daland. 2007. Linking grammatical principles with experimental speech production data: insights from Harmonic Grammar networks. Paper presented at Experimental Approaches to Optimality Theory, Ann Arbor, MI.
Goldwater, Sharon, and Mark Johnson. 2003. Learning OT constraint rankings using a maximum entropy model. In *Proceedings of the Workshop on Variation within Optimality Theory*, ed. by Jennifer Spenader, Anders Eriksson, and Östen Dahl, 111–120. Stockholm University.
Hayes, Bruce. 2004. Phonological acquisition in OT: the early stages. In *Fixing priorities: constraints in phonological acquisition*, ed. by René Kager, Joe Pater, and Wim Zonneveld, 158–203. Cambridge: Cambridge University Press. [ROA 327, 1999]
Hayes, Bruce, Bruce Tesar, and Kie Zuraw. 2003. OTSoft 2.1. Software package, http://www.linguistics.ucla.edu/people/hayes/otsoft/
Hinton, Geoffrey E., and Terrence J. Sejnowski. 1986. Learning and relearning in Boltzmann machines. In *Parallel distributed processing: explorations in the microstructure of cognition*, ed. by David E. Rumelhart, James L. McClelland, and the PDP Research Group, Vol. 1, 282–317. Cambridge, MA: MIT Press.
Jäger, Gerhard. To appear. Maximum entropy models and Stochastic Optimality Theory, In *Architectures, rules, and preferences: A festschrift for Joan Bresnan*, ed. by Jane Grimshaw, Joan Maling, Chris Manning, Jane Simpson, and Annie Zaenen. Stanford, CA: CSLI. [ROA-625, 2003]
Jäger, Gerhard, and Anette Rosenbach. 2006. The winner takes it all — almost: cumulativity in grammatical variation. *Linguistics* 44:937–971.
Jesney, Karen. 2007. The locus of variation in weighted constraint grammars. Poster presented at the *Workshop on Variation, Gradience and Frequency in Phonology*. Stanford University, July 2007. [Available at http://www.stanford.edu/dept/linguistics/linginst/nsf-workshop/Jesney_Poster.pdf]
Jesney, Karen, and Anne-Michelle Tessier. 2007a. Re-evaluating learning biases in Harmonic Grammar. In *University of Massachusetts Occasional Papers in Linguistics 36: Papers in theoretical and computational phonology*, ed. by Michael Becker. Amherst, MA: GLSA Publications.
Jesney, Karen, and Anne-Michele Tessier. 2007b. Restrictiveness in gradual learning of Harmonic Grammar. Paper presented at the North East Computational Phonology Workshop, University of Massachusetts, Amherst, November 10, 2007.
Johnson, Mark. 2002. Optimality-theoretic Lexical Functional Grammar. In *The lexical basis of syntactic processing: Formal, computational and experimental issues*, ed. by Suzanne Stevenson and Paola Merlo, 59–73. Amsterdam: John Benjamins.
Kager, René. 1999. *Optimality Theory*. Cambridge: Cambridge University Press.
Keller, Frank. 2000. *Gradience in grammar: experimental and computational aspects of degrees of grammaticality*. Doctoral dissertation, University of Edinburgh. [ROA-677]
Keller, Frank, and Ash Asudeh. 2002. Probabilistic learning algorithms and Optimality Theory. *Linguistic Inquiry* 33:225–244.
Keller, Frank. 2006. Linear Optimality Theory as a model of gradience in grammar. In *Gradience in grammar: Generative perspectives*, ed. by Gisbert Fanselow, Caroline Féry, Ralph Vogel, and Matthias Schlesewsky, 270–287. Oxford: Oxford University Press. [ROA-679]
Kiparsky, Paul. 1993. An OT perspective on phonological variation. Handout from *Rutgers Optimality Workshop 1993*, also presented at New Ways of Analyzing Variation (NWAV) 1994, Stanford University. [Available at http://www.stanford.edu/~kiparsky/Papers/nwave94.pdf].
Legendre, Géraldine, Yoshiro Miyata, and Paul Smolensky. 1990. Can connectionism contribute to syntax? Harmonic Grammar, with an application. In *Proceedings of the 26th Regional Meeting of the Chicago Linguistic Society*. ed. by M. Ziolkowski, M. Noske, and K. Deaton, 237-252. Chicago: Chicago Linguistic Society.
Legendre, Géraldine, Antonella Sorace, and Paul Smolensky. 2006. The Optimality Theory–Harmonic Grammar connection. In Smolensky and Legendre 2006, 903–966.
Levelt, Clara. 1995. Unfaithful kids: place of articulation patterns in early vocabularies. Colloquium presented at the University of Maryland, College Park.

Magri, Giorgio. 2007. The Multiplicative GLA. Paper presented at the Northeast Computational Phonology Workshop, University of Massachusetts, Amherst, November 10, 2007.
McCarthy, John J. 2002. *A thematic guide to Optimality Theory*. Cambridge: Cambridge University Press.
McCarthy, John J. 2008. *Doing Optimality Theory*. Malden, MA, and Oxford: Blackwell.
Nagy, Naomi, and Bill Reynolds. 1997. Optimality Theory and word-final deletion in Faetar. *Language Variation and Change* 9:37–55.
Novikoff, A.B.J. 1962. On convergence proofs on perceptrons. In *Proceedings of the Symposium on the Mathematical Theory of Automata* 12, 615–622. Polytechnic Institute of Brooklyn.
Ohala, Diane K. 1996. Cluster reduction and constraints in acquisition. Doctoral dissertation, University of Arizona.
Pater, Joe. 2008. Gradual learning and convergence. *Linguistic Inquiry* 39:334–345.
Pater, Joe, Rajesh Bhatt, and Christopher Potts. 2007. Linguistic optimization. Ms, University of Massachusetts, Amherst. [ROA-924]
Pater, Joe, Christopher Potts, and Rajesh Bhatt. 2007. Harmonic Grammar with linear programming. Ms, University of Massachusetts, Amherst. [ROA-872]
Prince, Alan. 2002. Anything goes. In *New century of phonology and phonological theory*, ed. by Takeru Honma, Masao Okazaki, Toshiyuki Tabata, and Shin-ichi Tanaka, 66–90. Tokyo: Kaitakusha. [ROA-536]
Prince, Alan. 2007a. The pursuit of theory. In *The Cambridge handbook of phonology*, ed. by Paul de Lacy, 33–60. Cambridge: Cambridge University Press.
Prince, Alan. 2007b. Let the decimal system do it for you: a very simple utility function for OT. Ms, Rutgers University. [ROA-943]
Prince, Alan, and Paul Smolensky. 1993/2004. *Optimality Theory: constraint interaction in generative grammar*. Technical Report, Rutgers University and University of Colorado at Boulder, 1993. Revised version published by Blackwell, 2004. [ROA-537]
Prince, Alan, and Paul Smolensky. 1997. Optimality: from neural networks to universal grammar. *Science* 275:1604–1610.
Prince, Alan, and Bruce Tesar. 2004. *Learning phonotactic distributions*. In *Fixing priorities: constraints in phonological acquisition*, ed. by René Kager, Joe Pater & Wim Zonneveld, 245–291. Cambridge: Cambridge University Press.
Reynolds, William. 1994. *Variation and phonological theory*. Doctoral dissertation, University of Pennsylvania, Philadelphia.
Riggle, Jason. 2004. *Generation, recognition and learning in finite state Optimality Theory*. Doctoral dissertation, UCLA.
Rosenblatt, Frank. 1958. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review* 65:386–408. [Reprinted in Anderson and Rosenfeld 1989]
Samek-Lodovici, Vieri, and Alan Prince. 1999. Optima. Ms, Rutgers University. [ROA-363]
Samek-Lodovici, Vieri, and Alan Prince. 2005. Fundamental properties of harmonic bounding. Ms, Rutgers University. [ROA-785]
Smolensky, Paul. 1986. Information processing in dynamical systems: foundations of Harmony Theory. In *Parallel distributed processing: explorations in the microstructure of cognition*, ed. by David E. Rumelhart, James L. McClelland, and the PDP Research Group, Vol. 1, 194–281. Cambridge, MA: MIT Press.
Smolensky, Paul. 1996. The initial state and 'Richness of the Base' in Optimality Theory. Ms, Johns Hopkins University. [ROA-154]
Smolensky, Paul. 2006. Optimality in phonology II: harmonic completeness, local constraint conjunction, and feature domain markedness. In Smolensky and Legendre, 27–160.
Smolensky, Paul, and Géraldine Legendre. 2006. *The harmonic mind: From neural computation to Optimality-Theoretic grammar*. Cambridge, MA: MIT Press.
Soderstrom, Melanie, Donald Mathis, and Paul Smolensky. 2006. Abstract genomic encoding of Universal Grammar in Optimality Theory. In Smolensky and Legendre, 403–471.
Tesar, Bruce. 1995. *Computational Optimality Theory*. Doctoral dissertation, University of Colorado, Boulder.
Tesar, Bruce. 1997. An iterative strategy for learning metrical stress in Optimality Theory. In *Proceedings of the 21st Annual Boston University Conference on Language Development*, ed. by Elizabeth Hughes, Mary Hughes, and Annabel Greenhill, 615–626. Somerville, MA: Cascadilla.

Tesar, Bruce. 2000. Using inconsistency detection to overcome structural ambiguity in language learning. Technical Report RuCCS-TR-58, Rutgers Center for Cognitive Science, Rutgers University, New Brunswick, NJ. [ROA-426]

Tesar, Bruce. 2007. A comparison of lexicographic and linear numeric optimization using violation difference ratios. Ms, Rutgers University. [ROA-939]

Tesar, Bruce, and Paul Smolensky. 1998. Learnability in Optimality Theory. *Linguistic Inquiry* 29:229–268.

Tesar, Bruce, and Paul Smolensky. 2000. *Learnability in Optimality Theory*. Cambridge, MA: MIT Press.

Tessier, Anne-Michelle. 2006. *Biases and stages in OT phonological acquisition*. Doctoral dissertation, University of Massachusetts, Amherst. [ROA-883]

Wexler, Kenneth, and Peter Culicover. 1980. *Formal principles of language acquisition*. Cambridge, MA: MIT Press.

Wilson, Colin. 2006. Learning phonology with substantive bias: an experimental and computational study of velar palatalization. *Cognitive Science* 30:945–982.

Yang, Charles. 2002. *Knowledge and learning in natural language*. New York: Oxford University Press.