Convergence Properties of a Gradual Learning Algorithm for Harmonic Grammar*

Paul Boersma and Joe Pater, 13 March 2013

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

The central question in research on language learnability in generative linguistics (e.g. Wexler and Culicover 1980, Dresher and Kaye 1990, Gibson and Wexler 1994, Fodor 1998, Tesar and Smolensky 1998, 2002, Yang 2002) is the following: is a given learning algorithm guaranteed to converge on a grammar that is correct for any language defined by a given theory of grammar? Research on language acquisition, on the other hand, studies the path taken to adult competence by human learners. Though clearly related, these two lines of research are often pursued completely independently of one another. In this paper, we investigate the convergence properties of a gradual learning algorithm for Harmonic Grammar (HG; Legendre, Miyata and Smolensky 1990/2006) and compare its behavior with similar learning algorithms developed for Optimality Theory (OT; Prince and Smolensky 1993/2004). An attractive feature of a gradual learning algorithm is its ability to model human learning paths (see e.g. Boersma and Levelt 2000 and Curtin and Zuraw 2002 on Boersma's 1998 GLA for stochastic OT, and Jäger 2007, Farris-Trimble 2008, Jesney and Tessier 2007a, 2011, and Jesney 2011 on the present GLA for HG). This paper strengthens the relationship between research on acquisition and learnability by showing that a gradual learning algorithm can also meet goals of learnability theory. By examining a learning algorithm that has its roots in neural modelling and statistical learning, this paper also strengthens the connections between these areas of research and the generative study of acquisition and learnability.

In section 2, we illustrate the HG model of grammar, and the learning algorithm, with a simple case involving stress placement. In section 3, we adapt the perceptron convergence proof to show that if the learner is given fully structured language data, this learning algorithm is guaranteed to find a correct set of constraint weights for any nonvarying language, i.e. any language in which each possible input to the grammar

generates the same output form on every occasion. The uniqueness of optima in the languages being learned means that, like most other generative learnability research, including Tesar and Smolensky's (1998, 2000) foundational work in OT, we operate under the idealization that languages do not display variation. In section 4, we test an implementation of the algorithm, and show that it quickly converges on a correct set of weights both for the original version of HG assumed in the convergence proof, and for Exponential HG, in which a restriction to positive values for weights is enforced through their exponentiation.

Section 5 extends the proof to a version of HG with probabilities on the weights, or noise, as in stochastic OT (Boersma 1998, Boersma and Hayes 2001). While we continue to operate with the idealization of nonvarying target languages, the addition of noise to the grammar increases its realism as a model of language acquisition, since the learner itself displays variation between grammatical "stages". The convergence of noisy HG learners on correct grammars for languages with unique optima is of further interest because the GLA for stochastic OT has been shown to fail on a case of this type (Pater 2008). In simulations, noisy HG and noisy Exponential HG learners converge on correct weights, while stochastic OT learners occasionally fail to converge.

The proofs and simulations in sections 3-5 operate under the idealization that learners have access to the full structure of the learning data. Section 6 presents simulations involving learning with hidden structure that use Tesar and Smolensky's (2000) Robust Interpretive Parsing. When applied to Tesar and Smolensky's test cases on the learning of stress, these simulations show that HG learners converge on correct grammars more frequently than OT learners with similar on-line learning algorithms (Error-Driven Constraint Demotion; Tesar and Smolensky 1998, 2000, Gradual Learning Algorithm; Boersma 1998, Boersma and Hayes 2001). We also find that convergence success rates are improved by noise on the constraint values. We use a simple example to show how symmetric update (promotion and demotion), noise, and weighted constraints can decrease the probability of a learner ending up in a trap caused by hidden structure.

2 Harmonic Grammar and its Gradual Learning Algorithm

We assume a model of grammar identical to that of OT except that it works with constraint ranks with constraint weights, as in OT's predecessor Harmonic Grammar (see Smolensky and Legendre 2006 and Pater 2009a for overviews of research in HG).

The tableaux in (1) illustrate HG evaluation of stress placement for a pair of words differing in syllable shape. Simplifying greatly, we assume that syllable structure is given in the underlying form, and only the placement of stress has to be determined. The first tableau (1a), then, is for an underlying /.ba.tan.ma./, which has a penultimate syllable with a coda /n/. We assume that this syllable is heavy, as opposed to the two CV syllables, which are light. Tableau (1a) determines where the stress lies in the phonological surface form, in which we simplify away the foot structure (see section 7 for an account *with* foot structure). If the second syllable is

stressed, as in the candidate surface form [.ba.tán.ma.], the constraint WEIGHT-TO-STRESS (W-TO-S; "every heavy syllable is stressed") is satisfied, which we indicate by an empty cell in the tableau. Placing stress instead on the final syllable, as in the candidate [ba.tan.má.], violates W-TO-S once, which we indicate with a score of −1 in the tableau (following Legendre, Sorace and Smolensky 2006 in regarding a violation as a negative satisfaction). This finally stressed alternative does satisfy the constraint ALIGN-R ("assign a violation to every syllable separating the stress from the right edge of the word"; McCarthy and Prince 1993), which the other candidate violates once, as indicated in the tableau.

In the tableaux in (1), the constraint weights (strengths) are shown beneath the constraint names. To the right of each row we see each candidate's *harmony*, which is calculated by multiplying its violation score on each constraint by that constraint's weight, and then summing the results. The optimal candidate is the one with the highest harmony, and is indicated by a pointing finger. The higher weight of W-TO-S (0.9) makes the score of the candidate that violates it (i.e. [.ba.tan.má.]) lower than one that violates ALIGN-R once (i.e. [.ba.tán.ma.]). Thus, the higher weight of W-TO-S as compared to ALIGN-R makes stress on the penultimate syllable optimal, just as its higher rank would do in OT.

(1) *Weighted constraint evaluation*

a. /.ba.tan.ma./	W-TO-S 0.9	ALIGN-R 0.6	
☞ [.ba.tán.ma.]		−1	−0.6
[.ba.tan.má.]	−1		−0.9
[.bá.tan.ma.]	−1	−2	−2.1

b. /.ban.ta.ma./	W-TO-S 0.9	ALIGN-R 0.6	
☞ [.ban.ta.má.]	−1		−0.9
[.bán.ta.ma.]		−2	−1.2
[.ban.tá.ma.]	−1	−1	−1.5

Tableau (1b) shows that the outcome is different when the heavy syllable is further from the right. The candidate surface form [.bán.ta.ma.] does satisfy the highest-valued constraint, W-TO-S, but its stressed syllable is two syllables away from the right edge; this candidate therefore incurs two violations of ALIGN-R, so that its harmony is $-2 \times 0.6 = -1.2$. This is a lower harmony than that of the alternative with final stress and an unstressed heavy first syllable, [.ban.ta.má.], which has a harmony of $-1 \times 0.9 = -0.9$ and is therefore optimal. This example illustrates the difference between HG's numerical measure of harmony and OT's ranking: with W-TO-S ranked above ALIGN-R, [.bán.ta.ma.] would be optimal in OT. For discussion of the typological consequences of the differences in outcomes with these constraints, see Legendre, Sorace and Smolensky (2006) and Pater (2009a, this volume).

The tableaux in (1) not only illustrate a *grammar* (with weights 0.9 and 0.6), but also a subset of a *language*, namely the set of underlying–surface pairs /.ba.tan.ma./–[.ba.tán.ma.] and /.ban.ta.ma./–[.ban.ta.má.]. As in other theories of grammar, the language that can be handled by the set of constraint weights is infinite, i.e. the grammar also applies to forms that are longer than the three syllables in (1): for forms with an arbitrary number of syllables of which exactly one is heavy, the grammar in (1) will place the stress on the penultimate syllable if it is heavy, otherwise on the final syllable. The goal of a human learner acquiring this language is to arrive at such constraint weights that her outputs, given the underlying forms /.ba.tan.ma./ and /.ban.ta.ma./, are precisely the surface forms [.ba.tán.ma.] and [.ban.ta.má.] (the correct outputs for longer forms then follow automatically). The learning algorithm discussed in this paper can now be illustrated informally. Suppose that at a certain point during the acquisition of her language a human learner has the grammar (constraint weights) in (2). She will produce the underlying form /.ba.tan.ma./ correctly as [.ba.tán.ma.], but the underlying form /.ban.ta.ma./ incorrectly as *[.bán.ta.ma.], as indicated in the tableau by a pair of asterisks around the pointing finger. The correct surface form [.ban.ta.má.] is indicated by a checkmark.

(2) *The learner*

a. /.ba.tan.ma./	W-TO-S 1.2	ALIGN-R 0.3	
☞ [.ba.tán.ma.]		−1	−0.3
[.ba.tan.má.]	−1		−1.2
[.bá.tan.ma.]	−1	−2	−1.8

b. /.ban.ta.ma./	W-TO-S 1.2	ALIGN-R 0.3	
√ [.ban.ta.má.]	−1		−1.2
☞ [.bán.ta.ma.]		−2	−0.6
[.ban.tá.ma.]	−1	−1	−1.5

Suppose now that the learner, having the grammar in (2), hears an adult produce the surface form [.ban.ta.má.]. If we assume that the learner can correctly infer the underlying form /.ban.ta.ma./ from this, we can conclude that the learner must notice that her grammar is in error, because she herself would produce a different surface form, namely [.bán.ta.ma.], from the given underlying form. The learner will then regard the second candidate in (2b) as her *incorrect optimum* (*☞*), and the first candidate as the *correct output* (√). Realizing that something must be wrong with the weight of the constraints W-TO-S and/or ALIGN-R (because these are the constraints that prefer either of the two forms), she can take action by improving the grammar, i.e. by changing one or both of these weights in an appropriate direction. Such an action is an example of error-driven learning: the learner is given a single correct pair of grammar inputs and outputs (e.g. a single correct underlying–surface pair) at a time, and adjusts the constraint weights only if 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 Gulicover 1980:127), from Error-Driven Constraint Demotion (EDCD) in OT (Tesar 1995), and from parameter setting algorithms (Gibson and Wexler 1994, Fodor 1998, Yang 2002).

The question is what precise action the learner should take. If she follows the learning algorithm discussed in this paper, she will both lower the weight of W-TO-S and raise the weight of ALIGN-R, making it more likely that she will turn an underlying /.ban.ta.ma./ into the correct output [.bán.ta.ma.] on a future occasion. In general, the Gradual Learning Algorithm for HG (HG-GLA) will raise the weight of each constraint that favours the correct output over the incorrect optimum by an amount ε (the *learning rate* or *plasticity*) multiplied by the number of violations by which that constraint favours the correct output, and it will lower the weight of each constraint that favours the incorrect optimum over the correct output by ε times the number of violations by which that constraint favours the incorrect optimum. So if ε = 0.1, then for the correct output [.ban.ta.má.] in (2b), the learner will lower the weight of W-TO-S by ε, changing it to 1.1, and she will raise the weight of ALIGN-R by 2ε, changing it to 0.5. This will not yet make the correct output the winner in her tableau for /.ban.ta.ma./, but it will do so after another two instances of [.ban.ta.má.] have arrived, which will change the weights to 1.0 and 0.6, respectively.

To make our discussion of learning more formal, we will represent the difference between the correct output and the incorrect optimum as an *error vector* (see also Pater this volume, where this is called a comparative vector). For tableau (2b) we get the error vector in (3). An error vector is obtained by subtracting the violation scores of the learner's incorrect optimal candidate from those of the desired correct output form. Error vectors are the HG equivalent of OT's winner-loser pairs (Tesar and Smolensky 1998, 2000; Prince 2003): positive scores indicate the degree to which a constraint prefers the correct output, and negative ones the degree of dispreference. For example, ALIGN-R prefers the correct [.ban.ta.má.] in (3) by 2 violations, and W-TO-S disprefers it by 1 violation.

(3) *Error vector*

/.ban.ta.ma./	W-TO-S	ALIGN-R
[.ban.ta.má.] – [.bán.ta.ma.]	−1	+2

From this error vector we can calculate how to update the weights. The error-driven learning algorithm we investigate updates each constraint's weight by multiplying its value on the error vector by a positive integer that is constant across constraints (the learning rate, or plasticity), and adding the result to its pre-update value (see section 3.1 for a formal statement). This update rule has been broadly applied in neural modelling and statistical learning under the names *perceptron update rule* (Rosenblatt 1958), *delta rule* (Widrow & Hoff, 1960), and *stochastic gradient ascent*. In generative linguistics, it is very similar to the update rule used in the gradual learning algorithm for OT in Boersma (1998) and Boersma and Hayes (2001) (see section 4.2 below on the difference), and it is identical to the update rules used in the connectionist implementation of HG in Soderstrom, Mathis and Smolensky (2006:

eqs. 14, 18, 21, 35d), the application of stochastic gradient ascent to a stochastic version of HG in Jäger (2007), and the application of the perceptron convergence procedure to categorical HG in Pater (2008). We will refer to the error-driven learning algorithm using this update rule with HG as HG-GLA (Harmonic Grammar's Gradual Learning Algorithm).

In the learning proofs and simulations that follow, we make the standard assumption that learners are exposed to a finite set of data sampled from the infinite set of pairs of grammar inputs and outputs (here, underlying and surface forms) that a language defines. The goal for a learning algorithm is to find a correct grammar with sufficient exposure to such learning data. Another standard assumption is that data that inform the learner that her grammar is still incorrect, such as [.ban.ta.má.] in the example above, are not "maliciously withheld" (Tesar and Smolensky 1998, 2000) from the learner (see section 3.3 for details). With this assumption about the availability of "informative data" we can derive learning paths for the example in (2). Figure 1 (left) shows the state in (2), with W-TO-S having a weight of 1.2 and ALIGN-R having a weight of 0.3, as a dot marked with "1". The dashed lines embrace the region where the learner should end up, namely the region where the weight of W-TO-S is greater than the weight of ALIGN-R (so that the grammar will generate [.ba.tán.ma] correctly) but less than twice the weight of ALIGN-R (so that the grammar will produce [.ban.ta.má.] correctly).

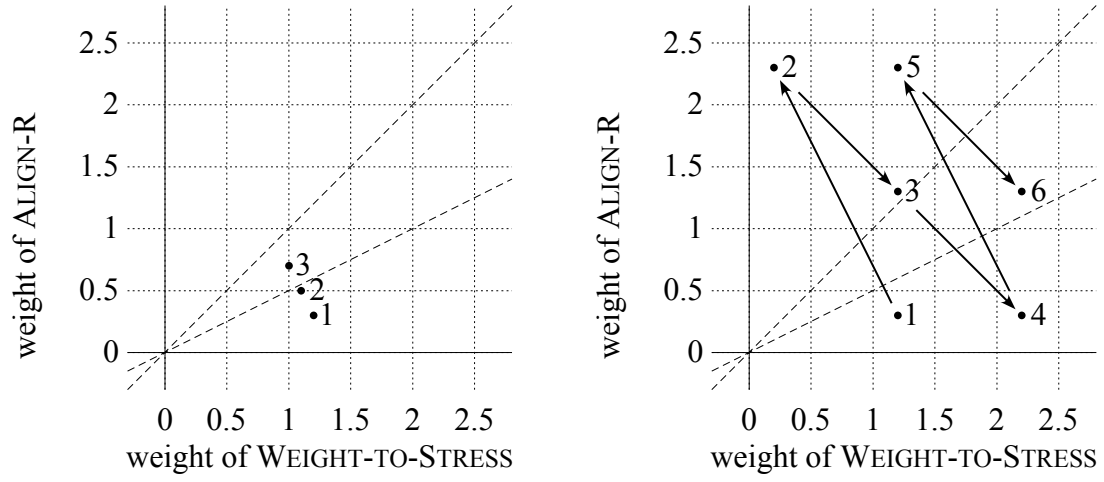

Fig. 1. Two learning paths from the same initial state:
at the left with a plasticity of 0.1, at the right with a plasticity of 1.0.

Let us follow the two learning paths shown in Figure 1. In the left figure, the plasticity is 0.1, as above, and the informative adult [.ban.ta.má.] moves the weights to 1.1 and 0.5, as above; this point is labelled as "2". The next [.ban.ta.má.] datum makes the learner move the weights to 1.0 and 0.7 (labelled as "3"), thus arriving in the target region. When her grammar is in the target region, she cannot make any errors any longer (her own optimum will always be identical to the incoming correct adult form), and learning stops forever. In other words, the learner has *converged* upon a correct grammar of the target language, in only two learning steps.

In the right-hand figure, the plasticity is 1.0, so that the first incoming adult [.ban.ta.má.] moves the weights to 0.2 and 2.3, labelled as "2". The learner has

thereby stridden past the target region. Fortunately, an incoming adult [.ba.tán.ma.] will now become become informative, as the learner's optimum will be the incorrect *[.ba.tan.má.]. From (2a) we can infer that the learner will change her weights to 1.2 and 1.3 (equal steps for both constraints, because the relevant error vector computed from (2a) has +1 for W-TO-S and −1 for ALIGN-R), labelled as "3" in the figure. Next, another incoming [.ba.tán.ma.] will move the weights to 2.2 and 0.3 (labelled as "4"), and a subsequent incoming [.ban.ta.má.] will move the weights to 1.2 and 2.3 ("5"), after which an incoming [.ba.tán.ma.] will move the weights to 2.2 and 1.3 ("6"), which is in the target region and is therefore the final state of the learner's grammar. In this case, convergence was reached after five learning steps. So we see that the learner eventually arrives in a correct grammar both for a small plasticity of 0.1 and for a large plasticity of 1.0.

It remains to be explained what happens if, as a rare coincidence, the learner arrives exactly on the boundary of the target region. In such a case, at least two candidates (one correct, the other incorrect) must be equally harmonic in the learner's grammar. The assumption about such a *tie* is that the learner will choose one of the tied candidates as her optimum at random. As the data come in repeatedly, the learner is sure to choose the incorrect candidate as her optimum at some point in time, and once she does so, she will notice her error and move the weights in a new learning step, moving away from the boundary.

Finally, there is the question of negative weights. It can be seen that if the plasticity is greater than both constraint weights, the first learning step in the above example will already move the weight of one of the constraints to a negative value. The learning algorithm will be seen to have no trouble with this situation.

With sufficient exposure to informative data, it turns out that an algorithm using the update rule mentioned above will always find a correct set of weights, independently of the initial weights of the constraints, independently of the plasticity, and independently of whether any weights ever become negative. This is shown in our adaptation of the perceptron convergence proof in sections 3 and 4, and is illustrated in our simulations of convergence times that follow the proofs. Specifically, these proofs show that for any nonvarying language (i.e. for any language in which every underlying form has a unique correct output) and for any set of constraints, an HG-GLA learner fed with a random sequence of correct underlying–surface pairs will need only a finite number of errors (learning steps) before it finds a correct grammar for the language.

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

As mentioned in section 2, no learning algorithm can be guaranteed to converge on a grammar that is appropriate for a target language unless it receives sufficient exposure

to informative data. Our proof therefore operates on the basis of a finite *dataset*, which contains a sufficient amount of informative data and from which the learning data that are supplied to the learner are sampled. This dataset, then, consists of a finite number M of "correct" *input-output* pairs. That is, the dataset has M different *input* (e.g. underlying) forms i_m (m = 1...M), each of which is coupled with a unique "correct" *output* (e.g. surface) form o_m. Note that we use the term "input" here in the OT sense of 'input to the grammar', not in the entirely different sense of 'language input to the learner' known from the language acquisition literature; here, the language input to the learner consists of both the inputs and the outputs of the grammar.

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 can change as a result of learning, as in Figure 1. We have five assumptions about the learner's processing, and three about learning.

The first assumption about processing, shared with existing proofs for EDCD and MaxEnt, 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 (1); this finiteness is required for the definition of the margin of separability in §3.2 and D_{max} in §3.3.[1] 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 denote the extent of constraint satisfaction by s_{mnk}; for instance, the value of s_{132} in tableaux (1) is −2 (first tableau, third row, second column). 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:

$$H_{mn} \equiv \sum_{k=1}^{K} w_k s_{mnk} \tag{4}$$

In (1), for instance, the value of H_{23} is −1.5. 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:

$$\forall n = 1..N_m : \quad \sum_{k=1}^{K} w_k s_{mrk} \geq \sum_{k=1}^{K} w_k s_{mnk} \tag{5}$$

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':

$$w_k' = w_k + \varepsilon \cdot \left(s_{pqk} - s_{prk} \right) \tag{6}$$

where ε is the plasticity, or learning rate, a positive real number. The formula is identical to that used by Jäger (2007) 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 of all the constraints 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 applying the perceptron convergence proof to HG-GLA 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 (it is possible, for instance, to construct languages in HG in which two candidates with different constraint violation patterns still tie for optimality; HG-GLA will typically not find a correct set of weights for such cases; our proof is therefore limited to target languages in which only a single candidate is correct for each tableau).

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 (e.g. any point between the two lines in Fig. 1), 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 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 (because they are the correct language forms, and the language is nonvarying). Therefore, the harmony (in the sample grammar) of any optimal form is greater than that of any other candidate for 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 the harmony comparison reads

$$\forall m = 1..M, n = 1..N_m, n \neq n_m: \quad \sum_{k=1}^{K} u_k s_{mn_m k} > \sum_{k=1}^{K} u_k s_{mnk} \tag{7}$$

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):

$$\forall m = 1..M, n = 1..N_m, n \neq n_m: \quad \sum_{k=1}^{K} u_k s_{mn_m k} \geq \sum_{k=1}^{K} u_k s_{mnk} + \delta \tag{8}$$

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 The Perceptron Convergence 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 (Rosenblatt 1962, Block 1962, Novikoff 1962, Minsky and Papert 1969; see Bishop 1995 for a textbook introduction). Our convergence proof for HG-GLA is closely analogous to the perceptron convergence proof by Block and Novikoff and many later formulations, including in linguistics (Collins 2002 applies it to part-of-speech tagging). In proving convergence for the update rule (7) in a constraint-weighting model, we are preceded by Fischer (2005), who provides a proof of its convergence for Maximum Entropy grammars; unfortunately, that 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 (7), but provided no convergence proof.

The simplest kind of perceptron proof would work for a single HG tableau with initial zero weights for all constraints; our proof here has to work from arbitrary initial weights (an element found in various existing perceptron proofs) and for any number of tableaus (a straightforward extension that also appears in e.g. Collins' proof). We will now show that the perceptron proof can be applied to HG-GLA because in §3.2 we managed to define the unit sample target grammar and the margin of separation of harmony.

The convergence proof for HG-GLA involves 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 Jesney and Tessier 2011 in HG).

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

$$w_k(t) = w_k(t-1) + \varepsilon \cdot \left(s_{p(t)q(t)k} - s_{p(t)r(t)k} \right) \tag{9}$$

We can now start off on some mathematical maneuvers that together allow us to establish an upper bound on the number of errors. First, we take the inner product of all terms in (10) with the weights of the unit sample target grammar:

$$\sum_{k=1}^{K} u_k w_k(t) = \sum_{k=1}^{K} u_k w_k(t-1) + \varepsilon \cdot \left(\sum_{k=1}^{K} u_k s_{p(t)q(t)k} - \sum_{k=1}^{K} u_k s_{p(t)r(t)k} \right) \tag{10}$$

With the help of (8), we can establish a lower bound on the last term, so that (10) becomes

$$\sum_{k=1}^{K} u_k w_k(t) \geq \sum_{k=1}^{K} u_k w_k(t-1) + \varepsilon\delta \tag{11}$$

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

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

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 (12) can be written as

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

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:

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

Since the first factor on the left in (14) is 1 (since u_k defines a unit grammar, its norm is 1), we can combine (13) and (14) into

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

Equation (15) 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, (15) 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 (15) is the norm of w_k. Equation (15) 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. 2):

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

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:

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

According to (4), the factor after 2ε involves a harmony difference:

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

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 (17) from above by throwing away the 2ε term from (17):

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

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 (19) can never be more than $8^2\varepsilon^2 K$. Generally, (19) becomes

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

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

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

The last term on the right in (21) is the squared norm of the learner's initial weight vector. We abbreviate it as $|W_0|^2$, so that (21) becomes

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

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. 2). But in (16) 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. 2), the two bounding conditions together necessarily indicate that t itself must be limited. More formally, we can combine (16) and (22) to yield

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

for $t \ge -W_0 \cdot U/(\varepsilon\delta)$. Equation (23) is a quadratic equation in t that can be solved:

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

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

The first thing to establish is that the right-hand side of this equation cannot be negative: if we write $t = 0$ in (14), and take into account that the norm of u_k is one, we see that $\left|W_0 \cdot U\right|$ must be less than or equal to $\left|W_0\right|$, so that the second term on the right in (25) must be positive or zero. We are therefore allowed to rewrite (25) as

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

Since (26) is valid for all $t \ge -W_0 \cdot U/(\varepsilon\delta)$, it is certainly valid for all $t \ge KD_{max}^2/(2\delta^2) - W_0 \cdot U/(\varepsilon\delta)$. For those values of t, (26) is equivalent to (27):

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

For smaller values of t, i.e. $t < KD_{max}^2/(2\delta^2) - W_0 \cdot U/(\varepsilon\delta)$, (26) is not equivalent to (27), but (27) holds a fortiori. Therefore, (27) expresses a true upper bound on t.

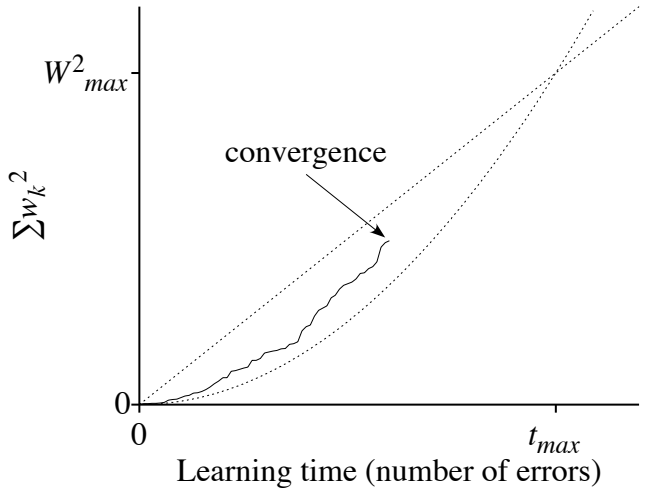

Fig. 2. A possible learning path (in terms of $\Sigma_k w_k^2$) if all weights start at zero. The path stays between the parabola of equation (16) and the line of equation (22). Convergence is reached well before $t_{max} = KD^2_{max}/\delta^2$. The value of W^2_{max} is $\varepsilon^2 K^2 D^4_{max}/\delta^2$.

The interpretation of (27) is that t can never be greater than a value t_{max} (also visible in Fig. 2) that is defined as the right-hand side of (27). 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 Constraint Demotion 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 the data are presented in an infinite sequence of sufficiently randomly selected input-output pairs. 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 (random selection was not addressed in the EDCD convergence proof by Tesar 1995, and therefore led to misconvergences for that algorithm; as here, the solution for EDCD lies in the random selection between ties, as shown by Boersma 2009).

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 (28), 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 Convergence Times: 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 could reasonably be a 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 (27) can be quite large. We address this issue by showing with computer simulations that the learning algorithm converges in a reasonable amount of time, for a number of randomly generated languages.

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 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 $\varepsilon = 1.0$.[2] 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 3 shows a histogram of the convergence times.

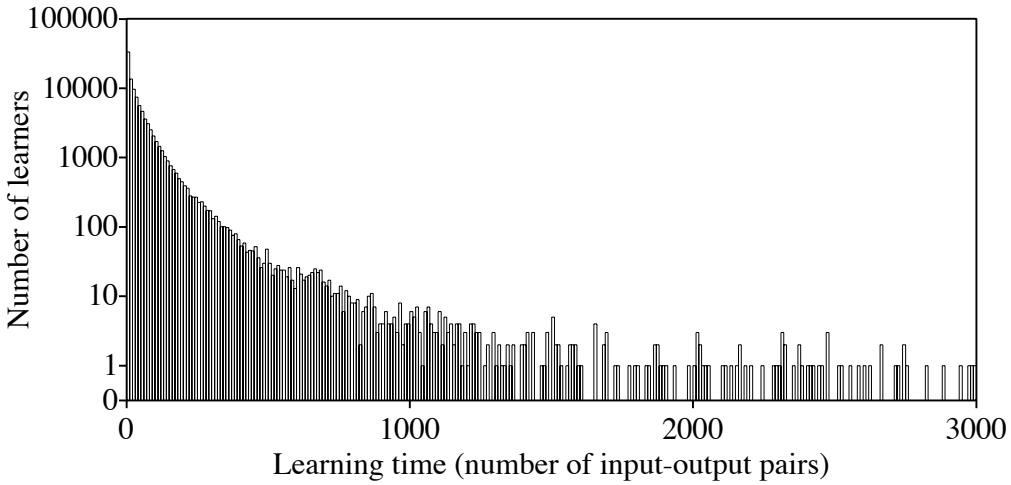

Fig. 3. Histogram of convergence times of 100,000 nonnoisy HG learners. The vertical scale is logarithmic.

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, although it takes them on average slightly more time than they need for the target HG grammars, perhaps because the weights have to be drawn further apart.

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 2009, Potts, Pater, Jesney, Bhatt and Becker 2010). To see why, consider the tableau in (28).

(28) *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): it can become optimal under no ranking of the constraints. 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).

One way to preserve OT-like harmonic bounding in HG, therefore, is not to use negative constraint 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 could use the *simplex* algorithm of linear programming, as shown by Potts *et al*. (2010); this method functions as the HG equivalent of the recursive version of Constraint Demotion, whose inconsistency detection properties are useful for calculating predicted typologies for OT (see Hayes, Tesar and Zuraw 2003 on typology calculations using recursive Constraint Demotion). The simplex algorithm is unlikely, however, 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 (Boersma and Weenink 1992–2013) since February 2006, the harmony of a representation is not given by (4), but by (29).

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

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 (6). Hence, the learning algorithm may shift the weight (w_k) of a constraint to a negative value; according to (29), 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.[3]

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"); we leave the full adaptation of this proof to grammar learning as a topic for further research. The learners are not only correct, but fast as well: Figure 4 shows

a histogram of the convergence times. The slowest learner needed 1160 input-output pairs to converge.

Exponential HG is thus a good candidate for adding OT-like behaviour to HG; it has OT-like harmonic bounding as well as a working learning algorithm. It can be mentioned here that another manner of banning negative weights, namely simply clipping all weights at zero from below (Keller 2000), also achieves 100% convergence in our simulations.

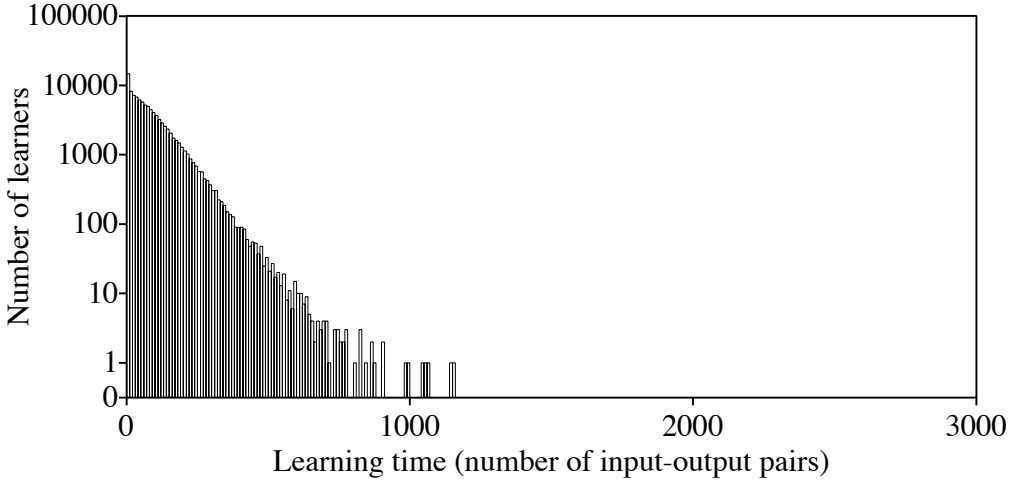

Fig. 4. Convergence times of 100,000 Exponential HG learners on OT-generated datasets.

4 Convergence of HG-GLA for Noisy Learners

Up to this point we have assumed a version of HG that is categorical in the sense that each time the same grammar is used to evaluate a candidate set, it returns the same candidate as optimal (or set of candidates, in the case of a tie). In both OT and HG, variation in the choice of optima across instances of evaluation has been widely used in accounting for natural language phenomena termed "free variation" (see Anttila 2007 and Coetzee and Pater 2011 for overviews). There are two extant stochastic versions of HG that yield this sort of variation. One stochastic variant of HG is the log-linear model of probabilistic grammar usually referred to as *Maximum Entropy grammar* (Johnson 2002, Goldwater and Johnson 2003, Fischer 2005, Wilson 2006, Jäger 2007). In this model, every output candidate has a relative probability that rises exponentially with its harmony. Another approach to stochastic HG is to incorporate the evaluation noise of stochastic OT (Boersma 1997; see also Ackley, Hinton and Sejnowski 1985 and Hinton and Sejnowski 1986 on noise on connection weights in neural networks, and Goldrick and Daland 2009 on a relationship of evaluation noise in HG to connectionist models of speech processing).

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 (30), 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}.

(30) *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

This kind of additive noise 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), Boersma and Escudero (2008), Pater (2009) and Coetzee and Pater (2011) 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 (Boersma 1998: 284; Boersma and Levelt 2000). Gradual learning with a categorical version of HG would produce an acquisition path with sudden changes from one optimum to another (e.g. sudden acquisition of codas, without a period of variation between deletion and accurate production).

Jäger (2007) shows that Maximum Entropy grammar with the HG-GLA update rule (that is, stochastic gradient ascent) yields the desired realistic learning curves. We here pursue the Noisy HG alternative because these two theories of stochastic HG have interesting differences in the patterns they produce. Maximum Entropy grammar, even a version that limits weights to positive values, always grants some probability to harmonically bounded candidates (Jäger and Rosenbach 2006). With the positivity restriction, these candidates can still never be granted the highest probability in their candidate sets. The empirical prediction is that there should be candidates that occur only in variation, but never as unique optima. In Noisy HG, however, such candidates have zero probability if the weights (including the evaluation noise) are limited to positive values (Jesney 2007; see here §3.5; for a boundary case, see Pater this volume Sec. 5). In this theory, the set of possible unique optima, and the set of candidates that can occur in variation, are identical. These differences in the theories of grammar are highly deserving of further investigation; here we aim to show that noisy HG is a viable alternative to Maximum Entropy grammar, and to stochastic OT, in terms of its convergence properties in combination with HG-GLA.

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

$$H_{mn} \equiv \sum_{k=1}^{K} \left(w_k + \mathbf{N}_k \right) s_{mnk} \tag{31}$$

where $\mathbf{N}_k$ is a Gaussian random variable. The lower bound on the sum of the squares of the constraint weights in (16) is still valid. The computation of the upper bound, however, changes, because (18) is no longer true. Instead, it is

$$\sum_{k=1}^{K} \left(w_k(t-1) + \mathbf{N}_k(t-1) \right) \left(s_{p(t)q(t)k} - s_{p(t)r(t)k} \right) = H_{p(t)q(t)}(t-1) - H_{p(t)r(t)}(t-1) \tag{32}$$

Thereby, (19) becomes

$$\sum_{k=1}^{K} w_k^2(t) \le \sum_{k=1}^{K} w_k^2(t-1) + \varepsilon^2 \sum_{k=1}^{K} \left(s_{p(t)q(t)k} - s_{p(t)r(t)k} \right)^2 - 2\varepsilon \sum_{k=1}^{K} \mathbf{N}_k(t-1) \left(s_{p(t)q(t)k} - s_{p(t)r(t)k} \right) \tag{33}$$

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 (33) is then $-N_{max}D_{max}$, so that an upper bound of the last sum is $2\varepsilon K N_{max} D_{max}$. Instead of (20) we thus get

$$\sum_{k=1}^{K} w_k^2(t) \le \sum_{k=1}^{K} w_k^2(t-1) + \varepsilon^2 K D_{max}^2 + 2\varepsilon K N_{max} D_{max} \tag{34}$$

In the end, convergence is guaranteed by

$$t \le \frac{\varepsilon K D_{max}^2 + 2K N_{max} D_{max}}{2\varepsilon\delta^2} - \frac{W_0 \cdot U}{\varepsilon\delta} + \sqrt{\left(\frac{\varepsilon K D_{max}^2 + 2K N_{max} D_{max}}{2\varepsilon\delta^2} - \frac{W_0 \cdot U}{\varepsilon\delta} \right)^2 + \frac{|W_0|^2 (W_0 \cdot U)^2}{\varepsilon^2\delta^2}} \tag{35}$$

The value for t_{max} that one can derive from (35) is, other than in (27), 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 $30N$ even during the whole 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 Convergence Times 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 5 shows a histogram of the convergence times.

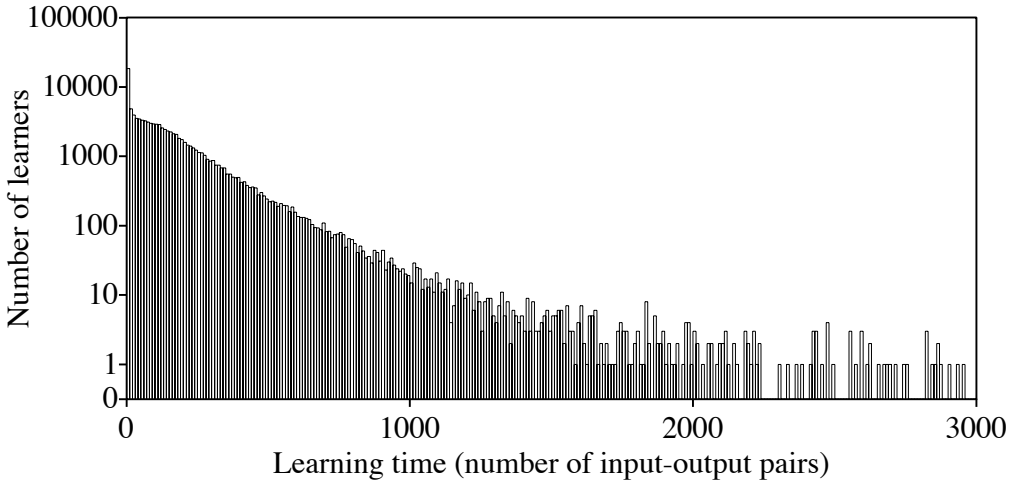

Fig. 5. Histogram of convergence times of 100,000 Noisy HG learners.

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 6 shows how the proficiency develops in time. This figure, then, shows this learner's *learning curve* for i_{19}.

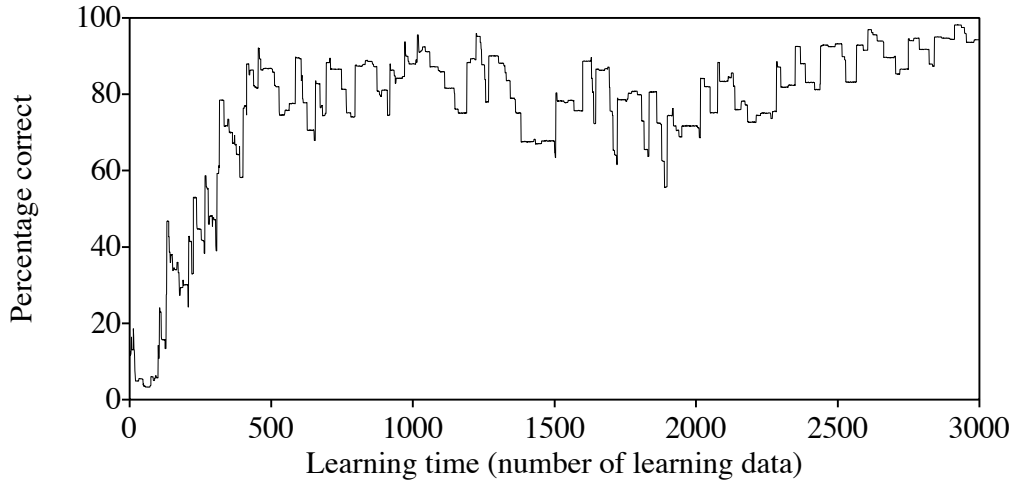

Fig. 6. The learning curve for one learner, for one input.

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 (30) with (32) and is given in (37).

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

In other words, noise is added to the "weights" w_k at evaluation time. Even if this sum results in a negative value, the subsequent exponentiation ensures that the violation score is multiplied by positive number. 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. 5, 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 (6) is replaced by the OT update rule in (37).

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

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 in HG the number of violations of a constraint with lower weight can matter, as illustrated in (1).

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)[4].

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. 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 (2007) Maximum Entropy learners (see also Coetzee and Pater 2011).

5 Learning with Imperfect Knowledge of Right and Wrong

In this section, we test the ability of HG-GLA to find a correct grammar when some of the linguistic structure is "hidden", using techniques and a benchmark learning problem developed by Tesar and Smolensky (2000). We then go on to discuss the differences in success amongst variants of HG-GLA and similar OT algorithms. To illustrate the general shape of the problem, we start with a much-discussed small portion of it: foot structure ambiguity for a medially stressed trisyllable.

In general, surface representations have more phonological structure than our examples in section 2. For instance, if a learner hears the overt phonetic form [batáma], she will not just assign the syllable structure [.ba.tá.ma.] to it (as in section 2), but also assign foot structure to it, and in this case the foot structure is ambiguous: [.ba.tá.ma.] can represent either the fully structured surface form [(ba.tá)ma.], with a right-headed foot (iamb) at the left edge of the word, or the fully structured surface form [.ba(tá.ma)], with a left-headed foot (trochee) at the right edge of the word.

Tableau 38 shows the relevant constraint violations for these two forms and several others. ALIGN-R and ALIGN-L are the constraints demanding that feet align with the right and left edge of the word respectively (McCarthy and Prince 1993), and TROCHEE and IAMB are the constraints demanding each kind of foot. The full structure in (38a) is consistent with a language in which a bisyllabic word is stressed on the final syllable (38e), because both require the same ranking or weighting of IAMB above TROCHEE. Similarly, the full structure in (38b) is consistent with a language in which a bisyllabic word is stressed initially (38f), since both satisfy TROCHEE.

(38) *The bata-batama universe*

Overt form	*Full structure*	ALIGN-R	ALIGN-L	TROCHEE	IAMB
[batáma]	a. [(ba.tá)ma.]	–1		–1	
	b. [.ba(tá.ma)]		–1		–1
[bátama]	c. [(bá.ta)ma.]	–1			–1
[batamá]	d. [.ba(ta.má)]		–1	–1	
[batá]	e. [(ba.tá)]			–1	
[báta]	f. [(bá.ta)]				–1

Given that foot structure is not directly marked in the phonetic signal, and that foot structure is needed for the learner to be able to assess constraint violations, it becomes relevant to ask the question of how a learner determines which full structure, (38a) or (38b), she has to assign to the incoming overt form [batáma]. Tesar (1997) and Tesar and Smolensky (2000) propose a general solution for such hidden structure problems called *robust interpretive parsing* (RIP), which fully stays within the spirit of OT and HG: of the two relevant candidate structures (38a) and (38b), choose the one that minimally violates the constraints as ranked (or weighted) in the learner's current grammar hypothesis. Tesar and Smolensky show that with this procedure, convergence is not guaranteed. Below we explain how RIP works, and how it sometimes fails, using an example from within the small universe of the two candidate sets in (38).

The first step in the learning procedure, robust interpretive parsing, uses the current grammar to assign the optimal full structure to an overt form. In our case of an

incoming overt [batáma], the learner will choose between (38a) and (38b) depending on which is preferred by the current state of the grammar. Let us assume that in the initial state all four constraints are ranked equally, as in (39a). Since both of the candidates violate two of the constraints, they are equivalently optimal.[5] We further assume that in a case of tie between optima, one is chosen randomly. Alongside each grammar in the learning sequence in (39) we include the overt learning data and the structures constructed by the learner that lead to a change in the state of the grammar. In (39a) we have overt [batáma], and interpretive parsing provides the fully structured surface form ("interpretation") [.ba(tá.ma)] – in (39) and elsewhere, the asterisk indicates that the choice was made from a set of tied optima. From this, the learner can recover the underlying form /batama/. Learning then proceeds in the general error-driven fashion described in earlier sections. In "production-oriented parsing", the learner finds the optimum from the entire candidate set for the underlying form /batama/, including now those that diverge from the overt form, which are (38c) and (38d). Since in our initial state all four candidates (38a–d) contain the same number of violations of equally ranked constraints, the choice is again made randomly, which could yield [(bá.ta)ma.], which is included in the production column of (39a). The full structure constructed in interpretive parsing ([.ba(tá.ma)]) does not match the result of production-oriented parsing ([(bá.ta)ma.]), so the learner updates the constraint ranks or weights. Error-Driven Constraint Demotion (EDCD; Tesar and Smolensky 1998, 2000) places any constraint that prefers the learner's incorrect optimum [(bá.ta)ma.] (here only ALIGN-L, as can be seen by comparing 38b with 38c) beneath the highest ranked constraint that prefers the allegedly correct form [.ba(tá.ma)] (here ALIGN-R). It will thus form the grammar in (39b), with ALIGN-L in a new lower stratum beneath ALIGN-R.

(39) *A path into a hidden structure trap for RIP/EDCD*

Grammar	**Overt**	**Interpret.**	**Production**
a. {ALIGN-L, ALIGN-R, TROCHEE, IAMB}	[batáma]	[.ba(tá.ma)]*	[(bá.ta)ma.]*
b. {ALIGN-R, TROCHEE, IAMB} >> ALIGN-L	[batá]	[(ba.tá)]	[(bá.ta)]*
c. {ALIGN-R, IAMB} >> {ALIGN-L, TROCHEE}	[batáma]	[.ba(tá.ma)]*	[.ba(ta.má)]
d. ALIGN-R >> {ALIGN-L, TROCHEE} >> IAMB	[batá]	[(ba.tá)]	[(bá.ta)]
e. ALIGN-R >> ALIGN-L >> IAMB >> TROCHEE	[batáma]	[.ba(tá.ma)]	[.ba(ta.má)]
f. ALIGN-R >> ALIGN-L >> TROCHEE >> IAMB	[batá]	[(ba.tá)]	[(bá.ta)]

Along with [batáma], our toy language includes bisyllabic [batá], which is presented to the learner next. The grammar in (39b) incorrectly judges [(ba.tá)] and [(bá.ta)] as equivalently optimal, so the learner may choose [(bá.ta)] in production and (after comparing 38e with 38f) demote TROCHEE beneath IAMB, as she has done in (39c). The new ranking now prefers iambs, which can lead to an error on [batáma], optionally interpretively parsed as [.ba(tá.ma)] under the tie between ALIGN-R and IAMB. Since production-oriented parsing picks iambic [.ba(ta.má)], IAMB will be demoted beneath TROCHEE. This ranking is now incorrect for [batá], and when the learner is presented with it, it will again demote TROCHEE beneath IAMB. The learner has now entered an endless cycle between the rankings in (39e) and (39f). The dominance of ALIGN-R over ALIGN-L and IAMB has removed the correct interpretive

parse [(ba.tá)ma.] from consideration, leaving the learner with access only to [.ba(tá.ma)], which is inconsistent with [(ba.tá)].

With a different set of random choices, EDCD would have succeeded on the above case. To assess how well RIP/EDCD performs on more realistic cases, Tesar and Smolensky (2000) tested it on similar metrical learning problems that involved 12 constraints and 62 overt forms for each of 124 languages. Here we extend Boersma's (2003) comparison of OT learners using EDCD and OT-GLA on these problems to also include HG and Exponential HG learners using HG-GLA. For each combination of grammar and learning algorithm, we also vary whether the grammar incorporates noise on the constraint values (except for EDCD, which is incompatible with noise). 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 (40).

(40) *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). HG-GLA learners turn out to have higher success rates than OT learners.[6]

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 (see below on the connection to local optima and simulated annealing). 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 (the real-life situation that this partial learnability corresponds to is

perhaps a language stage that is 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 (40), 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 (see also Apoussidou 2007: chs. 4, 5), and that a learner with an HG grammar has an advantage over one using OT-ranked constraints.

To show how symmetric update, noise, and weighted constraints can aid in hidden structure learning, we return to the *batá-batáma* example of (39), where a dominant ALIGN-R constraint caused a trap. Recall that when ALIGN-R determines the outcome of RIP, the resulting trochaic [.ba(tá.ma)] is inconsistent with iambic [(ba.tá)], which for EDCD leads to an inescapable cycle of demoting TROCHEE and IAMB beneath one another. A GLA learner, however, is not necessarily trapped by these circumstances. Take for example the learning sequence in (41), which results from the same data sequence and initial random choices that led EDCD into the trap in (39), but with the GLA update rule. Constraints have an initial value of 10, and the learning rate is 1; the results would be identical with either HG or OT as the model of grammar and either HG-GLA or OT-GLA as the learning algorithm. The first step yields the potentially dangerous dominance of ALIGN-R (41b). ALIGN-R has been promoted above the other constraints because the GLA update rule is symmetric: the values of the constraints preferring the form taken as correct are raised, alongside the lowering of the values of constraints preferring the learner's incorrect optimum.

(41) *A GLA learner with the same data sequence and initial random choices as (39)*

	ALIGN-L	**ALIGN-R**	**TROCHEE**	**IAMB**	**Overt**	**Interpret.**	**Product.**
a.	10	10	10	10	[batáma]	[.ba(tá.ma)]*	[(bá.ta)ma.]*
b.	9	11	10	10	[batá]	[(ba.tá)]	[(bá.ta)]*
c.	9	11	9	11	[batáma]	[.ba(tá.ma)]*	[.ba(ta.má)]
d.	9	11	10	10	[batá]	[(ba.tá)]	[(bá.ta)]*
e.	9	11	9	11	[batáma]	[(ba.tá)ma.]*	[.ba(ta.má)]
f.	10	10	9	11	[batáma]	[(ba.tá)ma.]	[.ba(ta.má)]
g.	11	9	9	11			

The next step brings IAMB up to the same level as ALIGN-R, since it prefers [(ba.tá)] over [(bá.ta)]. With a tie between IAMB and ALIGN-R, the GLA learner at (41c) is in the same position as the EDCD learner in (39c) of still allowing either [.ba(tá.ma)] or [(ba.tá)ma.] in interpretive parsing. As in (39c), we maliciously choose the problematic [.ba(tá.ma)]. The crucial difference is in what follows now. The EDCD learner in (39) went on to demote IAMB (and TROCHEE) to the bottom of the hierarchy,

thus rendering the desired [(ba.tá)ma.] subsequently inaccessible. The symmetric update rule, on the other hand, leaves TROCHEE and IAMB with the same average value as ALIGN-R and ALIGN-L, thus allowing IAMB to rise again to the same level as ALIGN-R. Eventually, the learner will randomly choose [(ba.tá)ma.] rather than [.ba(tá.ma)] in interpretive parsing, as in (41e). Once this happens, the learner is on the path toward the correct grammar, as shown in (41f) and (41g). In simulations, we find that a GLA with equal initial constraint values always converges on a correct grammar for *batá-batáma*, as opposed to the occasional failure of EDCD. We suspect that the difference between symmetric update and demotion-only that we have illustrated here is the cause of the better performance of the GLA learners found in the larger simulations.

It is possible to trap a GLA learner by placing ALIGN-R sufficiently high above the other constraints. An OT-GLA learner without evaluation noise will fail to converge if the distance between ALIGN-R and the other constraints is greater than the size of the learning step encoded in the learning rate. For example, with a learning rate of 1, the grammar in (42) is such a trap.

(42) *A hidden structure trap for a no-noise OT-GLA learner*

ALIGN-L	**ALIGN-R**	**TROCHEE**	**IAMB**
10	12	10	10

Starting with these constraint values, the noise-free OT-GLA learner will consistently pick [.ba(tá.ma)] in interpretive parsing, and never find a correct grammar. Adding noise to the grammar model changes this outcome because it gives some probability to [(ba.tá)ma.]. Once [(ba.tá)ma.] is chosen in interpretive parsing (and some other form is chosen in production), the constraint values are updated in its favour, and the learner is on its way out of the trap. In simulations, we find that the noise-free OT-GLA learner with the initial constraint values in (42) and a learning rate of 1 never converges on a correct grammar, while an OT-GLA learner with noise (standard deviation = 2) always does. More generally, adding a stochastic component to a model of learning or parsing is a commonly employed means of avoiding local minima (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 grammar in (42) is a local minimum for the noise-free OT-GLA learner because it yields the minimum error within the space of constraint values that the learner will explore.

The grammar in (42) is not a local minimum for an HG-GLA learner, even without noise. Since that grammar has a tie between IAMB and TROCHEE, the learner could make an error on [batá], leading to the grammar in (43). For the HG-GLA learner, this change has brought [(ba.tá)ma.] back into contention in interpretive parsing, since its harmony is now tied with [.ba(tá.ma)] – the weighted sum of violations of ALIGN-R and TROCHEE is the same as that of ALIGN-L and IAMB. For the OT-GLA learner, ALIGN-R still determines the outcome.

(43) *A step out of the trap in (42) for a no-noise HG-GLA learner*

ALIGN-L	**ALIGN-R**	**TROCHEE**	**IAMB**
10	12	9	11

In simulations, we find that in contrast with OT-GLA, a noise-free HG-GLA learner always succeeds when started with the weights in (42). The advantage of constraint weighting illustrated here is that there is a wider range of constraint values that lead to the result of [(ba.tá)ma.] being chosen as the optimal interpretation for overt [batáma]. We suspect that as in this example, HG learners generally more readily escape the potential local minima introduced by hidden structure because constraint weighting provides closer "stepping stones" out of these traps than ranking does. It is sometimes taken as a truism in generative linguistics that a learner is aided by minimizing the number of grammars that generate a given outcome. For hidden structure learning, this is at least sometimes false.

In sum, this section has shown that the performance of Tesar and Smolensky's (2000) procedure for learning with hidden structure, RIP, can be optimized by replacing demotion-only EDCD with a symmetric GLA, adding noisy evaluation, and replacing constraint ranking with weighting. We present these results as a useful baseline for further research on hidden structure learning. Even within the present approach of on-line learning with Harmonic Grammar, there are a number of avenues for further exploration, including changing the degree of noise over a training schedule (cf. simulated annealing), testing different models of stochastic HG, and different update rules. In research subsequent to Tesar and Smolensky (2000), Tesar (2000) proposes the Inconsistency Detection Learner, an algorithm for the learning of hidden structure that is guaranteed to converge, but at the cost of having to maintain multiple simultaneous grammar hypotheses; also, as Tesar notes (p. 37), it utterly fails when trying to 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.

6 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 expected to be robust against mistakes in the learning data (Boersma and Escudero 2008), and exhibits gradual learning curves; unlike in Stochastic OT, these attributes do not come at the cost of convergence failures. In addition to these core results, we have discussed an initial investigation into the hidden structure problem in HG learning, which provide some indications that there may be advantages for HG over OT in this domain. 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 – see Jesney this volume, Pater this volume, and the references cited therein, as well as Boersma and Escudero 2008 on HG's advantages with positive constraint satisfactions.

---

* Thanks to Diana Apoussidou, Rajesh Bhatt, Michael Collins, Chris Davis, Karen Jesney, Kie Zuraw and participants in Linguistics 794, Fall 2007 UMass for comments on earlier drafts. Thanks also to Gaja Jarosz, Mark Johnson, Giorgio Magri, Jason Naradowsky, Patrick Pratt, Robert Staubs and Colin Wilson for useful discussion, and to Bruce Tesar for supplying a list with the overt forms for the languages needed for the RIP simulations. This research was partially supported by grants 016.024.018 and 277-70-008 from the Netherlands Organization for Scientific Research (NWO) to Boersma and by grant BCS-0813829 from the National Science Foundation to the University of Massachusetts Amherst.

[1] Under some circumstances, a finite set of possibly optimal candidates might be derived from an infinite candidate set (e.g., Riggle 2004 on "contenders" in OT), and the proof could still work.

[2] In a computer implementation of a nonnoisy learner, the plasticity has to have an integer value, so as to avoid floating-point rounding errors, which 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).

[3] In a computer implementation, with its floating-point rounding errors, the trick of the previous footnote to handle ties no longer works. Instead, one can use a tiny evaluation noise in w_k, of e.g. 0.001, to force random choices between candidates with analytically "equal" harmonies.

[4] Magri (2012) fixes the GLA in such a way that it is guaranteed to converge for nonvarying data. Our preliminary tests with it, however, indicate that Magri's algorithm, unlike OT-GLA and HG-GLA, does not succeed with probabilistic data (see the next paragraph), so the search is still on.

[5] For compatibility with Tesar and Smolensky's (1998, 2000) original presentation, we here adopt their version of EDCD, which posits stratified hierarchies, in which tied constraints pool their violations. See Boersma (2009) for a revised EDCD that maintains a total order on the constraints, as in Prince and Smolensky (1993/2004). In that version, the learner could also fall into this trap through chance, selecting an unlucky random initial ranking of the constraints. In the simulations below, we use Boersma's (2009) EDCD, which unlike the original version, is guaranteed to converge on a correct total order in the absence of hidden structure. For results with the original version, see Tesar and Smolensky (2000) and Boersma (2003), but note that if the final strata of the "successful" learners in those papers were turned into totally ranked hierarchies, they would be incorrect half of the time, achieving lower success rates than we report here.

[6] Other learning algorithms can be considered. Jarosz (to appear) reports that an OT learner that randomly chooses a new ranking upon each error performs 100% correct on this test. However, the convergence time for this random algorithm scales exponentially with the number of independent constraints, so we do not consider this further here.

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.

Bishop, Christopher. 1995. *Neural networks for pattern recognition*. Oxford University Press.

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. 2008. 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]

Boersma, Paul, and David Weenink. 1992–2013. Praat, a system for doing phonetics by computer. http://www.praat.org

Coetzee, Andries, and Joe Pater. 2011. The place of variation in phonological theory. In *The handbook of phonological theory* (2nd ed.), ed. by John Goldsmith, Jason Riggle, and Alan Yu, 401–431. 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]

Curtin, Suzanne and Kie Zuraw. 2002. Explaining constraint demotion in a developing system. In *BUCLD 26: Proceedings of the 26th annual Boston University Conference on Language Development*, ed. by Anna H.-J. Do, Laura Domínguez, and Aimee Johansen. Cascadilla Press.

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]

Farris-Trimble, A. 2008. Cumulative faithfulness effects in phonology. Doctoral dissertation, Indiana University, xx.

Fischer, Markus. 2005. A Robbins-Monro type learning algorithm for an 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. 2007. Maximum entropy models and Stochastic Optimality Theory, In *Architectures, rules, and preferences: Variations on themes by Joan W. Bresnan*, ed. by Annie Zaenen, Jane Simpson, Tracy Holloway King, Jane Grimshaw, Joan Maling, and Chris Manning, 467–479. Stanford, CA: CSLI.

Jäger, Gerhard, and Anette Rosenbach. 2006. The winner takes it all — almost: cumulativity in grammatical variation. *Linguistics* 44:937–971.

Jarosz, Gaja. to appear. Naive parameter learning for Optimality Theory – The hidden structure problem. In *Proceedings of the 40th Annual Meeting of the North East Linguistic Society*.

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. 2011. Cumulative constraint interaction in phonological acquisition and typology. Doctoral dissertation, University of Massachusetts, Amherst.

Jesney, Karen. This volume. Positional constraints in Optimality and Harmonic Grammar.

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.

Jesney, Karen, and Anne-Michele Tessier. 2011. Biases in Harmonic Grammar: The road to restrictive learning. *Natural Language and Linguistic Theory* 29:251--290.

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.
Magri, Giorgio. 2012. Convergence of error-driven ranking algorithms. *Phonology* 29:213–269.
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.
Minsky, Marvin L. and Seymour A. Papert. 1969. Perceptrons. Cambridge, MA: MIT Press.
Nagy, Naomi, and Bill Reynolds. 1997. Optimality Theory and word-final deletion in Faetar. *Language Variation and Change* 9:37–55.
Novikoff, Albert 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. 2009a. Weighted constraints in generative linguistics. *Cognitive Science* 33: 999-1035 [ROA-982]
Pater, Joe. 2009b. Review of Smolensky and Legendre (2006). *Phonology* 26: 217-226. [ROA-983]
Pater, Joe. This volume. Universal Grammar with weighted constraints.
Potts, Christopher, Joe Pater, Karen Jesney, Rajesh Bhatt and Michael Becker. 2010. Harmonic Grammar with linear programming: from linear systems to linguistic typology. *Phonology* 27:77–117.
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]
Rosenblatt, Frank. 1962. *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*. Washington, DC: Spartan.
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.