Journal of Logic and Computation Advance Access published April 20, 2015

Noise robustness and stochastic tolerance of OT error-driven ranking algorithms

GIORGIO MAGRI*, *SFL UMR 7023, CNRS, Univ. Paris 8, 59/61 rue Pouchet, Paris 75017, France and UiL-OTS, Utrecht University, Trans 10, 3512 JK Utrecht, The Netherlands.*
E-mail: magrigrg@gmail.com

Abstract

Recent counterexamples show that Harmonic Grammar (HG) error-driven learning (with the classical Perceptron reweighing rule) is not robust to noise and does not tolerate the stochastic implementation (Magri 2014, MS). This article guarantees that no analogous counterexamples are possible for proper Optimality Theory (OT) error-driven learners. In fact, a simple extension of the OT convergence analysis developed in the literature (Tesar and Smolensky 1998, *Linguist. Inq.*, **29**, 229–268; Boersma 2009, *Linguist. Inq.*, **40**, 667–686; Magri 2012, *Phonology*, **29**, 213–269) is shown to ensure stochastic tolerance and noise robustness of the OT learner. Implications for the comparison between the HG and OT implementations of constraint-based phonology are discussed.

Keywords: Optimality Theory, error-driven learning, noise robustness, Harmonic Grammar.

In constraint-based phonology, grammars differ in how they regulate the conflict among phonological constraints. According to *Optimality Theory* (OT; [28]), grammars differ in how they prioritize the constraints into a ranking. According to *Harmonic Grammar* (HG; [17, 18, 30]), grammars differ in how they weight the constraints when averaging their corresponding violation numbers. A (gradual) *error-driven learner* in OT or HG slightly re-ranks or re-weights the constraints whenever an error is detected on the current piece of data.[1] In a realistic learning setting, a fraction of the training data might have been corrupted by (production or transmission) *noise* ([12, p. 410]; [11, p. 625]; [6, pp. 66–67]; [2]). Various authors have also considered an implementation of error-driven learning which is called *stochastic* because the current ranking or the current weights are stochastically 'perturbed' at every iteration ([3, 4, 6–10, 14]).

Is the learner robust to noise or is it instead the case that a few pieces of noisy data can require a large number of updates to recover? Does the learner tolerate the stochastic implementation or is it instead the case that the stochastic component can disrupt efficient learning? Recent counterexamples show that the HG error-driven learner currently used in the literature (based on the Perceptron reweighing rule) is not robust to noise and does not tolerate the stochastic implementation ([24]). This article provides analytical guarantees that no analogous counterexamples are possible for a properly crafted OT error-driven learner. In fact, a simple twist of the OT convergence analysis

[1]A *gradual* error-driven learner updates the current grammar only 'slightly', so that a single piece of data can trigger multiple consecutive updates because a single update does not necessarily suffice to make the current grammar consistent with that piece of data. A *non-gradual* learner instead performs updates large enough that the current grammar becomes instantaneously consistent with the current piece of data, so that no piece of data can trigger two consecutive updates. The non-gradual learner can be analysed as the gradual variant run on a training sequence where the same piece of data is fed to the learner multiple consecutive times, until it cannot trigger any further updates. For instance, Magri [21, section 3.7] reinterprets accordingly the non-gradual EDCD algorithm developed in [31]. The analysis of gradual error-driven learning thus trivially extends to the non-gradual variant. For this reason, I focus on the gradual implementation throughout the article.

doi:10.1093/logcom/exv011

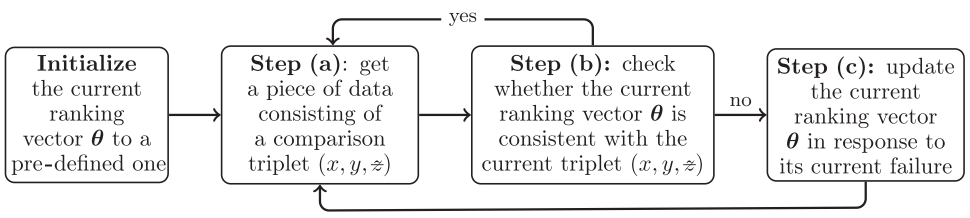
developed in the literature ([5, 21, 31]) is shown to ensure noise robustness and stochastic tolerance of the OT error-driven learner. This article thus contributes to the comparison between the HG and OT implementations of constraint-based phonology from the perspective of error-driven learnability.

The article is organized as follows. Section 1 reviews error-driven learning within OT. Section 2 briefly reviews the OT convergence analysis and error bounds. The most general case is considered, whereby the re-ranking operation performed by the OT learner involves both the demotion of 'faulty' constraints and the promotion of 'virtuous' constraints. Section 3 extends those convergence results and error bounds to the stochastic implementation, thus providing guarantees that the OT learner tolerates the stochastic component. Section 4 extends convergence results and error bounds to a noisy learning setting, thus providing guarantees that the OT learner is robust to noise. Section 5 discusses the quality of the error bounds and their dependence on the promotion component of the re-ranking rule. Section 6 concludes with a discussion of the HG counterexamples mentioned above from the perspective of the OT results obtained in Sections 3 and 4.

1 Error-driven ranking algorithms

In this article, I focus on an implementation of error-driven learning in OT whereby the learner entertains a numerical representation of its current hypothesis on the target grammar ([3, 4]). Concretely, the OT error-driven learner assigns to each constraint C_k a numerical *ranking value* θ_k. These values are collected into a tuple $\boldsymbol{\theta} = (\theta_1, \theta_2, \dots)$ called a *ranking vector*. The current ranking vector is updated as schematized in (1).

(1)


yes
Initialize the current ranking vector $\boldsymbol{\theta}$ to a pre-defined one
Step (a): get a piece of data consisting of a comparison triplet $(x, y, \not\approx)$
Step (b): check whether the current ranking vector $\boldsymbol{\theta}$ is consistent with the current triplet $(x, y, \not\approx)$
no
Step (c): update the current ranking vector $\boldsymbol{\theta}$ in response to its current failure

To start, the current ranking values are initialized to some pre-defined values. For concreteness, I assume throughout that the current ranking values are all initialized to zero. The analyses reported in this article easily extend to arbitrary initial ranking values ([31, appendix A.4]; [21, appendix A]). After initialization, the learner enters a loop that consists of three steps. At step (1a), the learner is fed a piece of data. At step (1b), the learner checks whether that piece of data is consistent with the current ranking vector. If consistency holds, the learner has nothing to learn from the current piece of data, loops back to step (1a) and waits for more data. If instead consistency fails, the learner takes action by updating the current ranking vector at step (1c), and then starts all over again. The rest of this section specifies the nature of the data fed at step (1a), the notion of consistency checked at step (1b), and the update rule used at step (1c).

At a minimum, the piece of data fed to the learner at step (1a) consists of a surface form y, say some form which is licit according to the phonotactics corresponding to the target grammar the learner is being trained on. In some applications, the corresponding underlying form x can be assumed to be provided as well. In other applications instead, the learner needs to be endowed with an additional subroutine to reconstruct the underlying form. Yet, the analyses reported in this article are independent of the subroutine for the choice of the underlying form x, which I thus assume to be provided in some arbitrary way along with a surface form y at step (1a). The mapping (x, y) of the underlying form x to the surface form y must, therefore, beat the mapping $(x,\cancel{z})$ of x to any other loser candidate $\cancel{z}$ (loser candidates are stricken out as a mnemonic) according to the target grammar the learner is being trained on. The learner needs to focus on one such loser candidate $\cancel{z}$. Usually, this loser candidate is chosen through a proper subroutine. Yet, the analyses reported in this article are independent of the subroutine for the choice of the loser form $\cancel{z}$, which I thus assume to be provided in some arbitrary way at step (1a) as well. In the end, the piece of data fed to the learner at step (1a) consists of a *comparison triplet* $(x,y,\cancel{z})$ which pits an intended winner candidate y against an intended loser candidate $\cancel{z}$ relative to an underlying form x.

At step (1b), the learner checks whether the current ranking vector is *consistent* with the current comparison triplet $(x,y,\cancel{z})$. That triplet partitions the constraints into three types: the *winner-preferring* constraints, which assign less violations to the winner mapping (x,y) than to the loser mapping $(x,\cancel{z})$; the *loser-preferring* constraints, which instead assign less violations to the loser mapping $(x,\cancel{z})$ than to the winner mapping (x,y); and the *even* constraints, which assign the same number of violations to the two mappings. A constraint C_k which is loser-preferring relative to the current triplet is called *undominated* relative to the current ranking vector $\boldsymbol{\theta}=(\theta_1,\theta_2,\ldots)$ provided its current ranking value θ_k is at least as large as the largest current ranking value among the winner-preferring constraints, as stated in (2), where W stands for the set of winner-preferring constraints.[2]

(2) $\theta_k \geq \max_{h\in W} \theta_h$

The current triplet is *consistent* with the current ranking values provided there exist no undominated loser-preferring constraints; if any exists, consistency fails.

If consistency fails, the current ranking vector $\boldsymbol{\theta}$ needs to be updated at step (1c). The update should move the learner closer to the aim of making any loser-preferring constraint dominated. The update thus consists of two operations. On the one hand, the update demotes each undominated loser-preferring constraint C_k by decreasing its ranking value θ_k by a small *demotion amount*, so that the left-hand side of (2) becomes smaller. On the other hand, the update promotes each winner-preferring constraint C_h by increasing its ranking value θ_h by a small *promotion amount*, so that the right-hand side of (2) becomes larger ([31]; [4, p. 323–327]; [21]). The actual sizes of the promotion and demotion amounts do not matter, only their ratio does. The demotion amount is thus set equal to 1 for concreteness while the promotion amount is allowed to take on an arbitrary value, leading to the update rule (3).

(3) a. Decrease by 1 the ranking value of each undominated loser-preferring constraint.

b. Increase the ranking value of each winner-preferring constraint by a certain *promotion amount*.

The promotion amount in (3b) is allowed to be null, in which case the learner only performs constraint demotion and no promotion (see below for an example).

The learner is said to *converge* provided it only performs a finite number of updates. This means that it eventually settles on a *final ranking vector* $\boldsymbol{\theta}=(\theta_1,\theta_2,\ldots)$ whereby the loser-preferring constraints

[2]The maximum over an empty set is defined as equal to $-\infty$. If the set W of winner-preferring constraints is empty, condition (2) thus boils down to $\theta_k \geq -\infty$ and is, therefore, always satisfied. In other words, every loser-preferring constraint counts as undominated if there are no winner-preferring constraints.

of each triplet are dominated and consistency is thus ensured. Consider any constraint ranking $\gg$ which *respects* the order implicitly defined by the relative size of the final ranking values, in the sense of condition (4). This condition says that, if a certain constraint has a larger final ranking value than some other constraint, the former is ranked on top of the latter by the ranking $\gg$. If instead the two constraints have the same final ranking value, then the ranking $\gg$ can split the tie either way.

(4) If $\theta_h > \theta_k$, then $C_h \gg C_k$.

Since the final ranking vector $\boldsymbol{\theta}$ is consistent with any training triplet and since the constraint ranking $\gg$ respects the final ranking vector, the constraint ranking $\gg$ must rank a winner-preferring constraint on top of every loser-preferring constraint for any training triplet. This conclusion means in turn that the OT grammar corresponding to that constraint ranking $\gg$ ensures that the winner candidate y beats the loser candidate z relative to the underlying form x for any training triplet (x, y, z). The learner has, therefore, managed to learn a constraint ranking which accounts for all the training data. For this reason, the learner just described is called an *error-driven ranking algorithm* (EDRA) for OT.

Let me close this section with a concrete example. Suppose that the constraint set consists of just three constraints: a markedness constraint $M = *$[+VOICE, −SONORANT] against voiced obstruents together with general and positional faithfulness constraints $F_{\text{gen}} =$ IDENT[VOICE] and $F_{\text{pos}} =$ IDENT[VOICE]/ONSET for voicing. Suppose that the learner is trained on the two comparison triplets (5) and (6), which sort the constraints into winner-preferring, loser-preferring, and even as indicated. These training data provide evidence that obstruent voicing is neutralized in coda but not onset position.

(5) a. (/da/, [da], ~~[ta]~~)

b.

F_{pos} = IDENT[VOICE]/ONSET:	winner-preferring
M = *[+VOICE]:	loser-preferring
F_{gen} = IDENT[VOICE]:	winner-preferring

(6) a. (/rad/, [rat], ~~[rad]~~)

b.

F_{pos} = IDENT[VOICE]/ONSET:	even
M = *[+VOICE]:	winner-preferring
F_{gen} = IDENT[VOICE]:	loser-preferring

Ranking vectors consist of three rankings values, with the convention that the first value corresponds to the constraint F_{pos}, the second to M, and the third to F_{gen}. Suppose that the EDRA adopts the re-ranking rule (3) with the promotion amount always set equal to zero for concreteness. The complete learning dynamics is described in (7).

(7)

$$\begin{tikzcd}
& \begin{bmatrix}0\\-1\\0\end{bmatrix} \arrow[dr, "(/\text{rad}/, [\text{rat}], \cancel{[\text{rad}]})"] & & \\
\begin{array}{c|c} F_{\text{pos}} & \\ M & \begin{bmatrix}0\\0\\0\end{bmatrix} \\ F_{\text{gen}} & \end{array} \arrow[ur, "(/\text{da}/, [\text{da}], \cancel{[\text{ta}]})"] \arrow[dr, "(/\text{rad}/, [\text{rat}], \cancel{[\text{rad}]})"'] & & \begin{bmatrix}0\\-1\\-1\end{bmatrix} \arrow[r, "(/\text{rad}/, [\text{rat}], \cancel{[\text{rad}]})"] & \begin{bmatrix}0\\-1\\-2\end{bmatrix} \\
& \begin{bmatrix}0\\0\\-1\end{bmatrix} \arrow[ur, "(/\text{da}/, [\text{da}], \cancel{[\text{ta}]})"'] & &
\end{tikzcd}$$

At the first iteration, the EDRA can receive either of the two training triplets. For concreteness, suppose it receives the triplet (/da/, [da], ~~[ta]~~), as in the upper path of (7). Since that triplet is not consistent with the initial ranking vector (the loser-preferring constraint M is undominated), the EDRA demotes by 1 the ranking value of the undominated loser-preferrer M. At the next iteration, the EDRA can again receive either of the two training triplets. Since the current ranking vector is already consistent with the triplet (/da/, [da], ~~[ta]~~), no update is performed until the EDRA is fed the triplet (/rad/, [rat], ~~[rad]~~) instead. Since the latter triplet is not consistent with the current ranking vector, the EDRA demotes by 1 the ranking value of the currently undominated loser-preferrer F_{gen}. And so on.

No matter the training sequence, the current raking vector entertained after three updates is consistent with both training triplets. Learning thus ceases and converge holds. The final ranking vector in (7) assigns the largest ranking value to constraint F_{pos}, an intermediate value to M, and the smallest value to F_{gen}. There is therefore a unique constraint ranking that respects the relative size of the final ranking values, namely the one in (8a). This constraint ranking indeed makes the winner in the two triplets beat the corresponding loser according to OT, as shown by the tableaux in (8b).

(8) a.

$$\begin{tikzcd} F_{\text{pos}} \arrow[d, dash] \\ M \arrow[d, dash] \\ F_{\text{gen}} \end{tikzcd}$$

b.

/rad/	F_{pos}	M	F_{gen}
a. ☞ [rat]			$*$
b. [rad]		$*!$	

/da/	F_{pos}	M	F_{gen}
a. ☞ [da]		$*$	
b. [ta]	$*!$		$*$

At convergence, the EDRA has thus succeeded at finding a solution (8a) to the problem of learning an OT grammar which accounts for the training data.

2 Review of the theory of EDRAs' convergence

Consider again the example illustrated at the end of the preceding section. Constraint F_{pos} is ranked at the top of the target ranking (8a) and its ranking value never goes below 0 in the learning dynamics (7). In fact, constraint F_{pos} can be ranked at the top only because it is never loser-preferring according to the training triplets (5)–(6) and, therefore, never demoted from its initial ranking value equal to zero. Constraint M is ranked next in the target ranking (8a) and its ranking value never goes below -1 in the learning dynamics (7). In fact, M can be ranked second because it is only loser-preferring relative to the comparison triplet (/da/, [da], ~~[ta]~~) where the top-ranked constraint F_{pos} is winner-preferring. In order for that triplet to trigger an update, its loser-preferring constraint M must be undominated, namely must be currently ranked above the winner-preferrer F_{pos}. Since the ranking value of F_{pos} is always zero, M can only be demoted when its current ranking value is zero. Hence, M cannot ever make it below -1. Finally, constraint F_{gen} is ranked third according to the target ranking (8a) and its ranking value never goes below -2 in the dynamics (7). In fact, F_{gen} needs to be ranked third because it is loser-preferring relative to the comparison triplet (/rad/, [rat], ~~[rad]~~) whose winer-preferrer M is ranked second. In order for this triplet to trigger an update, its loser-preferring constraint F_{gen} must be undominated, namely must be currently ranked above the winner-preferrer M. Since the ranking value of M can only drop down to -1, F_{gen} can only be demoted when its current ranking value is equal to 0 or to -1. Hence, F_{gen} cannot ever make it below -2. This reasoning holds in complete generality: if the EDRA is trained on triplets that are consistent with a ranking that assigns a certain constraint to the k-th slot, the ranking value of that constraint can never become smaller than $-(k-1)$, as stated in the following Lemma 1 (due to [31]; see also [5]). The proof is recalled in Appendix A for completeness.

Lemma 1
Suppose that an EDRA is trained on a sequence of comparison triplets consistent with some constraint ranking. Without loss of generality, assume that ranking is $C_1 \gg C_2 \gg \ldots \gg C_k \gg \ldots$, whereby constraint C_k is assigned to the k-th slot (the first slot being the top of the ranking). The ranking value θ_k of constraint C_k entertained by the EDRA at an arbitrary iteration can be bounded as follows:

(9) $\theta_k \geq -(k-1)$

so that the sum of the current ranking values can be bounded as follows:

(10) $\theta_1 + \theta_2 + \ldots + \theta_n \geq -\frac{1}{2}n(n-1)$

where n is the total number of constraints. ■

The promotion component (3b) of the re-ranking rule promotes the winner-preferring constraints by a certain *promotion amount*, which can be null or positive. Express the promotion amount as in (11), namely as the inverse of the number of winner-preferring constraints, scaled by a (non-negative) constant. This constant thus calibrates the promotion amount on the number of winner-preferring constraints, and it is, therefore, called the *calibration* constant. It can be zero, in which case the promotion amount is null and the learner performs no constraint promotion.

(11) $\text{Promotion amount used in (3b)} = \dfrac{\text{calibration constant}}{\text{number of winner-preferrers}}$

To illustrate, consider an update triggered by the comparison triplet (/da/, [da], ~~[ta]~~) described in (5). The loser-preferring constraint M is demoted by 1. Furthermore, each of the two winner-preferring constraints F_{pos} and F_{gen} is promoted by half of the calibration constant according to (11). In the end, the sum of the current ranking values is decreased by subtracting 1 (because of the demotion) and increased by adding the calibration constant (because of the two promotions each equal to half the calibration constant). Because of the shape (11) of the promotion amount, this conclusion holds in the general case: any update modifies the sum of the current ranking values by an amount equal to ($\textit{calibration} - 1$). Since the initial ranking values are equal to zero, the sum of the current ranking values after a certain number of updates can be bounded as in (12).[3]

(12) $\theta_1 + \theta_2 + \ldots + \theta_n \leq \text{number of updates} \times \big(\textit{calibration} - 1\big)$

To illustrate, consider the run (7), corresponding to a null promotion amount, namely one with the shape (11) with the calibration constant equal to zero. In that case, (12) says that the sum of the current ranking values after T updates cannot be larger than $-T$. And that is indeed the case: the sum of the current ranking values after $T=1$ updates is equal to -1; after $T=2$ updates, it is equal to -2; and after $T=3$ updates at the end of the run, it is equal to -3.

Suppose that the constant in the numerator of (11) used to calibrate the promotion amount is strictly smaller than 1. Hence, the coefficient ($\textit{calibration} - 1$) which appears in (12) is strictly negative. In

[3]In (12), we have an inequality rather than an identity because an update can demote a number ℓ of undominated loser-preferring constrains, so that the sum of the current ranking values changes by ($\textit{calibration} - \ell$).

that case, the two inequalities (10) and (12) can be combined into the bound on the number of updates stated in (13) (due to [31] for the demotion-only case and extended to a non-null promotion amount in [20, 21]). This bound grows slowly (quadratically) with the number n of constraints. It thus ensures that the EDRA converges and that converge is efficient when the number of errors is compared with the number of constraints which need to be ranked.

THEOREM 1
Express the promotion amount in terms of the number of winner-preferring constraints through a calibration constant, as in (11). If the calibration constant is strictly smaller than 1, the EDRA converges: whenever trained on comparison triplets consistent with some OT grammar, the number of updates can be bounded as follows

$$\text{(13)}\quad \text{Number of updates} \leq \frac{1}{2}\frac{1}{1-\text{calibration}}n(n-1)$$

where n is the total number of constraints. ■

If the calibration constant is equal to 1, the learner converges but efficiency is lost: it is possible to construct cases where the number of updates grows exponentially with the number of constraints ([21]). If the calibration constant is larger than 1, convergence is lost as well: it is possible to construct cases where the number of updates is infinite ([26]). We have thus obtained a sufficient and necessary characterization of EDRA's efficient convergence in terms of the condition that the calibration constant be strictly smaller than 1.

3 Stochastic tolerance

The implementation of error-driven learning considered so far is called *deterministic*, to distinguish it from the *stochastic* implementation. The two implementations only differ because of which loser-preferring constraints count as undominated. In the case of the deterministic EDRA, a loser-preferring constraint C_k is undominated relative to the current ranking values $\theta_1, \theta_2, \ldots$ if it satisfies condition (2). In the case of the stochastic EDRA, it counts as undominated if it satisfies the alternative condition (14), where the values $\epsilon_1, \epsilon_2, \ldots$ are sampled at every iteration independently of each other according to an arbitrary but fixed underlaying distribution ([3, 4, 6–10, 14]).

$$\text{(14)}\quad \theta_k + \epsilon_k \geq \max_{h \in W}(\theta_h + \epsilon_h)$$

The stochastic condition (14) coincides with the original condition (2) only applied not to the current ranking values $\theta_1, \theta_2, \ldots$ but to their stochastic counterpart $\theta_1+\epsilon_1, \theta_2+\epsilon_2, \ldots$. In other words, the original condition (2) checked by the deterministic EDRA declares a loser-preferring constraint C_k undominated provided its current ranking value θ_k is not smaller than the current ranking value θ_h of some winner-preferring constraint C_h. The alternative condition (14) checked by the stochastic EDRA declares a loser-preferring constraint C_k undominated provided its stochastic ranking value $\theta_k+\epsilon_k$ is not smaller than the stochastic ranking value $\theta_h+\epsilon_h$ of some winner-preferring constraint C_h.

The stochastic EDRA can make more errors than the deterministic one, because the stochastic component can derail the learner away from the most straightforward learning path. The worst case number of errors made by the stochastic EDRA can thus be expressed as the sum of two terms. The first term (15a) is the number of errors made by the deterministic EDRA, which measures the difficulty of error-driven learning. The second term (15b) is the number of *additional* errors made by the stochastic EDRA, which measures the slowdown due to the stochastic implementation.

(15) $$\text{Number of errors made by the } \textit{stochastic} \text{ EDRA} = \underbrace{\text{Number of errors made by the } \textit{deterministic} \text{ EDRA}}_{\text{(a)}} + \underbrace{\text{Number of additional errors due to the stochastic implementation}}_{\text{(b)}}$$

Do EDRAs *tolerate* the stochastic implementation? In other words, is the number (15b) of additional errors finite so as not to disrupt convergence? Is it furthermore small enough not to disrupt efficiency? This section provides a positive answer to these questions and thus concludes that error-driven learning in OT tolerates the stochastic implementation.

Consider a constraint C_k which is loser-preferring according to the current comparative triplet. The deterministic and the stochastic definitions (2) and (14) of undominatedness are repeated for comparison in (16a) and (16b), respectively, slightly rearranged. The value ϵ which appears on the right-hand side of (16b) is the difference $\epsilon = \epsilon_k - \epsilon_H$ between the stochastic value ϵ_k corresponding to the loser-preferring constraint C_k minus the stochastic value ϵ_H corresponding to the winner-preferring constraint C_H which has the largest stochastic ranking value among winner-preferrers, namely $\theta_H + \epsilon_H = \max_{h \in W} (\theta_h + \epsilon_h)$.

(16) a. Deterministic undominatedness:
$$\theta_k - \max_{h \in W} \theta_h > 0$$
b. Stochastic undominatedness:
$$\theta_k - \max_{h \in W} \theta_h > \epsilon$$

The deterministic condition (16a) and the stochastic condition (16b) only differ because zero on the right-hand side of the former has been replaced in the latter by a certain value ϵ. The latter value ϵ could be different from zero. Yet, assume that the stochastic values $\epsilon_1, \epsilon_2, \ldots$ have been sampled according to a distribution bounded between $-\Delta$ and $+\Delta$, for some threshold Δ. Then, the right-hand side ϵ of (16b) cannot be much different from zero, since it is the difference between two stochastic values which are in turn bounded between $-\Delta$ and $+\Delta$. Since the deterministic and stochastic implementations only differ for the definition of undominatedness and since the two definitions are very similar, Lemma 1 for the deterministic EDRA extends to the following analogous Lemma 2 for the stochastic EDRA. The two lemmas only differ because of the additional term (17b) in the bound on the current ranking values entertained by the stochastic EDRA. This additional term is intuitively due to the fact that the stochastic EDRA might happen to demote loser-preferring constraints which, although undominated relative to the stochastic condition (14)/(16b), are nonetheless not undominated relative to the original deterministic condition (2)/(16a). In the worst case, the stochastic EDRAs can thus demote the constraints further down. The proof of Lemma 2 is a straightforward variant of Tesar and Smolensky's proof of Lemma 1, and is provided in Appendix B for completeness.

Lemma 2
Suppose that a stochastic EDRA is trained on a sequence of comparison triplets consistent with some constraint ranking. Without loss of generality, assume that ranking is $C_1 \gg C_2 \gg \ldots \gg C_k \gg \ldots$, whereby constraint C_k is assigned to the k-th slot (the first slot being the top of the ranking). Suppose furthermore that the stochastic values ϵ_k are always bounded between $-\Delta$ and $+\Delta$, for some threshold $\Delta \geq 0$. The ranking value θ_k of constraint C_k entertained by the stochastic EDRA at an arbitrary

iteration can be bounded as follows:

$$(17) \quad \theta_k \geq -\underbrace{(k-1)}_{(a)} \underbrace{-2\Delta(k-1)}_{(b)}$$

so that the sum of the current ranking values can be bounded as follows:

$$(18) \quad \theta_1+\theta_2+\ldots+\theta_n \geq -\frac{1}{2}n(n-1)-\Delta n(n-1)$$

where n is the total number of constraints. ■

The inequality (12) used in the analysis of the deterministic EDRA was independent of the definition of undominatedness and instead only followed from the choice of a promotion amount that depends on the number of winner-preferring constraints through a calibration constant as in (11). Since the difference between the deterministic and the stochastic EDRA only concerns the notion of undominatedness, the inequality (12) extends straightforwardly to the stochastic case. Combining the latter inequality (12) with the inequality (18) provided by the lemma yields the bound (19) on the number of updates performed by the stochastic EDRA.

THEOREM 2
Assume that the stochastic values $\epsilon_1, \epsilon_2, \ldots$ are always bounded between $-\Delta$ and $+\Delta$, for some threshold $\Delta \geq 0$. Express the promotion amount in terms of the number of winner-preferring constraints through a calibration constant, as in (11). If the calibration constant is strictly smaller than 1, the stochastic EDRA converges: whenever trained on comparison triplets consistent with some OT grammar, the number of updates can be bounded as follows

$$(19) \quad \text{Number of updates} \leq \underbrace{\frac{1}{2}\frac{1}{1-\text{calibration}}n(n-1)}_{\text{(a)}} + \underbrace{\frac{\Delta}{1-\text{calibration}}n(n-1)}_{\text{(b)}}$$

where n is the total number of constraints that the stochastic EDRA re-ranks. ■

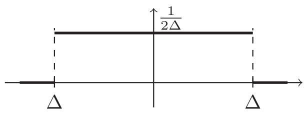
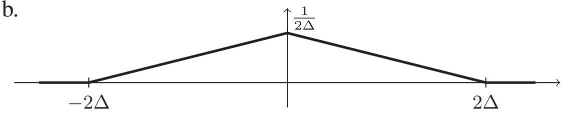
The error bound (19) has the expected shape (15): the term (19a) coincides with the error bound (13) for the deterministic EDRA and the term (19b) thus expresses the number of additional errors due to the stochastic implementation. This additional term (19b) grows slowly (quadratically) with the number n of constraints.[4] Theorem 2 thus guarantees that EDRAs tolerate the stochastic implementation. The additional term (19b) in the error bound is due to the additional term (17b) in the lower bound on the current ranking values.

Theorem 2 holds under the assumption that the stochastic values ϵ_k are deterministically bounded between $-\Delta$ and $+\Delta$ for some threshold Δ which in turn appears in the bound (19). The literature instead usually assumes that these stochastic values ϵ_k are sampled according to a Gaussian distribution with zero mean and small variance ([3, 4, 6]). This assumption allows the stochastic

[4] The number (15b) of additional errors due to the stochastic component also depends on the 'size' of the stochastic component. Lemma 2 and Theorem 2 assume that the stochastic values $\epsilon_1, \epsilon_2, \ldots$ are sampled between $-\Delta$ and $+\Delta$. Under this assumption, the size of the stochastic component is the constant Δ. In fact, as Δ grows, the stochastic ranking values $\theta_k + \epsilon_k$ are allowed to get further apart from their deterministic counterpart θ_k, so that the stochastic component exerts a larger effect. The bound (19b) thus says that the number of additional updates due to the stochastic implementation grows slowly (linearly) with the size Δ of the stochastic component.

values ϵ_k to be smaller than $-\Delta$ or larger than $+\Delta$. Yet, it still guarantees that the stochastic values ϵ_k are bounded between $-\Delta$ and $+\Delta$ with large probability.[5] Theorem 2 thus holds also when the stochastic values ϵ_k are sampled according to a Gaussian distribution, although it holds not deterministically but only with high probability (a probability which in turn depends on the threshold Δ used in the bound).

The choice between a Gaussian distribution and one bounded between $-\Delta$ and $+\Delta$ has little implications not only from the computational but also from the modelling perspective. The simplest bounded distribution is the *uniform distribution* $\mathcal{U}(-\Delta, +\Delta)$, whose probability density function is plotted in (20a): it is constant between $-\Delta$ and $+\Delta$ and equal to zero elsewhere. This uniform distribution is quite different from a Gaussian distribution. Yet, the difference $\epsilon_k - \epsilon_h$ between two uniform random variables ϵ_h, ϵ_k is distributed according to the *triangular* probability density function plotted in (20b): it is largest at zero, decreases linearly, and is zero at the left of -2Δ and at the right of 2Δ.

(20) a.  b. 

Crucially, the stochastic values ϵ_k never enter into the definition of the algorithm individually, but only through the difference $\epsilon = \epsilon_k - \epsilon_H$ which appears on the right-hand side of the definition (16b) of stochastic undominatedness. It is the distribution of ϵ which matters for the modelling predictions of the learner, not the distribution of the individual values ϵ_k. Assuming that the individual values ϵ_k have a uniform distribution ensures that the difference ϵ has a triangular distribution, which is very close to the Gaussian distribution.

In conclusion, this section has established that error-driven learning in OT tolerates the stochastic implementation. This conclusion is best appreciated through comparison with error-driven learning in the related framework of HG, based on the *Perceptron* reweighing rule (or a truncated variant thereof, which ensures that the weights are non-negative; [25]). Magri [24] shows that this HG error-driven learner does not tolerate the stochastic implementation. In fact, the number of additional errors due to the stochastic component can grow so fast with the number of constraints that a counterexample with just 10 constraints is shown to force the HG stochastic learner to make over 1 million *additional* errors. On that same test case, the stochastic OT learner is instead reported to make less than 150 additional errors. This sharp contrast between the HG and the OT learners holds despite the fact that the violation profiles in these counterexamples have been chosen in such a way that the HG and OT typologies explored by the two learners exactly coincide. Theorem 2 backs up the positive OT simulation results with formal guarantees: it says that the OT mode of constraint interaction has special formal properties that support the stochastic implementation, as opposed to the HG mode of constraint interaction.

4 Noise robustness

So far, I have assumed that the learner is trained on a sequence of *pristine* comparative triplets all consistent with some ranking (vector). A more realistic learning setting needs instead to allow for

[5]In fact, the probability that a value ϵ_k sampled according to the normal distribution with variance σ lies between $-\Delta$ and $+\Delta$ can be expressed as $\mathrm{erf}(\Delta/(\sigma\sqrt{2}))$. The function $\mathrm{erf}(x)$ is very close to 1 already for small values of x (for instance, $\mathrm{erf}(3.5) = 0.999999$).

the possibility that a fraction of the training data could have been *corrupted* by transmission noise or speech errors, leading to training triplets which are inconsistent with the pristine ones ([12, p. 410]; [11, p. 625]; [6, pp. 66-67]; [2]). No assumptions are made here on the corrupted comparative triplets apart from there being only a finite number of them. Indeed, if an infinite number of corrupted triplets were allowed, the worst case number of errors would always be infinite: whenever the learner rested on a current ranking vector, we could prompt it to perform yet another update by maliciously crafting a proper piece of corrupted data. The corrupted training triplets make the learning problem harder ([22, section 2]) and might force the learner to make more errors. In other words, the number of errors made by the learner on a possibly corrupted training sequence can be expressed as the sum of two terms. The first term (21a) is the number of errors that would have been made on the sequence of pristine data with the corrupted data removed. The second term (21b) is the number of *additional* errors due to the corrupted data.

$$(21)\quad \begin{array}{c}\text{Number of errors}\\ \text{made by the EDRA}\\ \text{in the noisy setting}\end{array} = \underbrace{\begin{array}{c}\text{Number of errors}\\ \text{made in the}\\ \text{noise-free case}\end{array}}_{\text{(a)}} + \underbrace{\begin{array}{c}\text{Number of additional errors}\\ \text{due to the corrupted}\\ \text{training data}\end{array}}_{\text{(b)}}$$

Are EDRAs robust to nose? In other words, is the number of additional errors (21b) due to the noise corrupted data small enough not to disrupt efficiency? This section provides a positive answer to this question and thus concludes that error-driven learning in OT is robust to noise. It is only for simplicity that the analysis of the noisy learning setting is limited to the deterministic EDRA in this section. In fact, the analysis could be easily combined with the analysis of the stochastic EDRA in the preceding section, yielding error bounds for the stochastic EDRA in the noisy setting.

Lemma 1 for the noise-free setting extends to Lemma 3 for the noisy setting. The two lemmas only differ because of the additional term (22b) in the bound on the current ranking values entertained by the EDRA. The latter additional term is intuitively due to the fact that constraint C_k is demoted not only by the pristine but also by the corrupted triplets. The amount of demotion caused by the pristine triplets can be bounded independently of their number, through the assumption that they are consistent, leading to the term (3a). The amount of demotion caused by the corrupted triplets can only be bounded through the only assumption we have on those triplets, namely that there is only a finite number δ_h of them which can demote a generic constraint C_h, leading to the term (3b). The proof of lemma 3 is a straightforward variant of Tesar and Smolensky's proof of lemma 1, and is provided in Appendix C for completeness.

Lemma 3

Suppose that the EDRA's training sequence consists of both *pristine* and *corrupted* training triplets. The pristine triplets are all consistent with some constraint ranking. Without loss of generality, assume that ranking is $C_1 \gg C_2 \gg \ldots \gg C_k \gg \ldots$, whereby constraint C_k is assigned to the k-th slot (the first slot being the top of the ranking). The corrupted triplets are instead inconsistent with this ranking $C_1 \gg C_2 \gg \ldots \gg C_k \gg \ldots$ and thus count as noise. Assume that the training sequence contains only a finite number of these corrupted triplets (while the pristine triplets can be infinite in number), and let δ_h be the number of corrupted triplets in the training sequence where constraint C_h is loser-preferring. The current ranking value θ_k of constraint C_k entertained by the deterministic EDRA at an arbitrary iteration can be bounded as follows:

$$(22)\quad \theta_k \geq \underbrace{-(k-1)}_{(a)} - \underbrace{\sum_{h=1}^{k} \delta_h}_{(b)}$$

so that the sum of the current ranking values can be bounded as follows:

$$(23)\quad \theta_1 + \theta_2 + \ldots + \theta_n \geq -\frac{1}{2} n(n-1) - n(n-1)\delta$$

where n is the number of constraints and δ is the number of corrupted triplets in the training sequence. ■

The inequality (12) used in the analysis of the noise-free setting follows from the fact that the sum of the current ranking values decreases by $(calibration - 1)$ with each update. That follows in turn from the definition of the re-ranking rule (3) and in particular from the choice of a promotion amount with the form (11). In other words, the inequality (12) holds no matter whether the update has been triggered by a pristine or by a corrupted triplet and, therefore, extends to the noisy learning setting considered here. The two inequalities (23) and (12) can then be combined into the error bound (24).

THEOREM 3
Express the promotion amount in terms of the number of winner-preferring constraints through a calibration constant, as in (11). If the calibration constant is strictly smaller than 1, the deterministic EDRA converges also in the noisy learning setting: when trained on a sequence of *pristine* triplets consistent with some OT grammar interspersed with a finite number δ of *corrupted* triplets inconsistent with the pristine ones, the number of updates can be bounded as follows:

$$(24)\quad \text{Number of updates} \leq \underbrace{\frac{1}{2}\frac{1}{1-\text{calibration}} n(n-1)}_{\text{(a)}} + \underbrace{\delta \frac{1}{1-\text{calibration}} n(n-1)}_{\text{(b)}}$$

where n is the number of constraints. ■

The error bound (24) has the expected shape (21): the term (24a) coincides with the error bound (13) for the noise-free setting and the term (24b) thus expresses the number of additional errors due to the corrupted training data. The latter term (24b) is in turn the product between the number δ of corrupted data times a factor which thus quantifies the amount of disruption caused by each corrupted piece of data. This additional term (24b) grows slowly (quadratically) in the number n of constraints (and only linearly in the number δ of corrupted data). Theorem 3 thus guarantees that the EDRA is robust to noise.

This conclusion is best appreciated through comparison with error-driven learning in the related framework of HG, based on the *Perceptron* reweighing rule (or a truncated variant thereof, which ensures that the weights are non-negative; [25]). Magri [24] shows that this HG error-driven learner is not robust to noise. In fact, the number of additional updates needed to recover from a single faulty update triggered by a single noise corrupted triplet can grow so fast with the number of constraints that a counterexample with just 10 constraints is shown to require the HG learner to make over 30,000 *additional* updates in order to recover from a single faulty update. On that same test case, the OT learner makes only nine additional updates to recover from the faulty update triggered by that piece of noisy data. This sharp contrast between the HG and the OT learners hold despite the fact that the violation profiles in these counterexamples have been chosen in such a way that the HG and OT typologies explored by the two learners exactly coincide. Theorem 3 backs up the positive OT simulation results with formal guarantees: it says that the OT mode of constraint interaction has special formal properties that support error-driven learning also in a realistically noisy learning setting, as opposed to the HG mode of constraint interaction.

5 Dependence of the error bounds on the promotion amount

The three error bounds (13), (19) and (24) have so far been discussed from the perspective of their dependence on the number n of constraints. These error bounds also depend on the constant used to calibrate the promotion amount through (11). This calibration constant is bounded between 0 (included) and 1 (excluded). It enters into the error bounds through the multiplicative factor $1/(1-\mathit{calibration})$. This factor is equal to 1 when the calibration constant is smallest (namely equal to 0) and goes to infinity as the calibration constant grows (and approaches 1). In other words, the error bounds are optimal (namely smallest) when the calibration constant is null and the EDRA thus performs only constraint demotion but no constraint promotion. The error bounds get worse (namely larger) as the calibration constant grows closer to the forbidden threshold of 1 and the EDRA thus performs more and more constraint promotion. Is the growth of the error bound with the calibration constant due to a weakness of the analysis? In the sense that an improved (namely smaller) bound is possible for the promotion/demotion learner? Or is it indeed the case that constraint promotion indeed slows down the learner in the general case? This section addresses this question.

Consider again the example presented at the end of section 1. The EDRA was trained on the two comparative triplets (/da/, [da], ~~[ta]~~) and (/rad/, [rat], ~~[rad]~~) described in (5) and (6). And it had to re-rank the three constraints F_{pos}, M, and F_{gen}. The comparative triplets sort the constraints into winner- and loser-preferring constraints as summarized in the matrix (25). This matrix has two rows, corresponding to the two triplets. It has three columns, corresponding to the three constraints. The entries of the matrix are W’s or L’s, depending on whether the corresponding constraint is winner- or loser-preferring relative to the corresponding triplet (even constraints correspond to blank entries).

(25)
$$\begin{array}{r|ccc|} & F_{\text{pos}} & M & F_{\text{gen}} \\ (\text{/da/, [da], }\cancel{\text{[ta]}}) & \text{W} & \text{L} & \text{W} \\ (\text{/rad/, [rat], }\cancel{\text{[rad]}}) & & \text{W} & \text{L} \end{array}$$

In order to compute the learning dynamics (7), we do not really need to know the identity of the underlying/winner/loser forms in each triplet or the definition of the constraints. The abstract matrix (25) of W’s and L’s suffices, as the consistency condition checked by the EDRA at step (1b) and the re-ranking operation executed at step (1c) are entirely defined in terms of winner- and loser-preferring constraints. In general, a test case for the EDRA can thus be described without spelling out the triplets and the constraints, simply through a matrix whose rows correspond to implicit triplets, whose columns correspond to implicit constraints, and whose entries are W’s and L’s and thus specify the winner- or loser-preferring constraints. Each row of the matrix is called an *elementary ranking condition* (ERC; [27]).

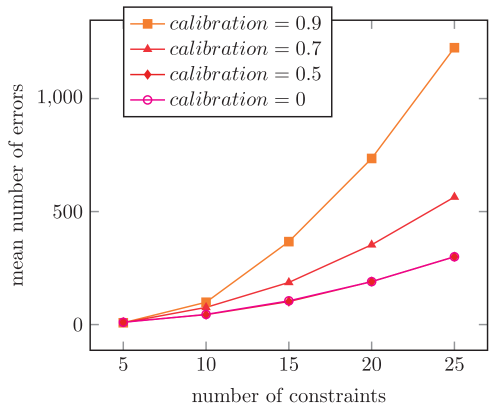
Pater [26] considers ERC matrices with n columns and $n-1$ rows which consist of a diagonal layer of W’s, followed by a layer of L’s, followed by another layer of W’s. To illustrate, Pater’s matrices

(*P-matrices*) corresponding to $n = 4, 5, 6$ constraints are listed in (26).

(26)
$$
\begin{array}{cccc}
C_1 & C_2 & C_3 & C_4 \\
\hline
\mathrm{W} & \mathrm{L} & \mathbf{W} & \\
 & \mathrm{W} & \mathrm{L} & \mathbf{W} \\
 & & \mathrm{W} & \mathrm{L}
\end{array}
\qquad
\begin{array}{ccccc}
C_1 & C_2 & C_3 & C_4 & C_5 \\
\hline
\mathrm{W} & \mathrm{L} & \mathbf{W} & & \\
 & \mathrm{W} & \mathrm{L} & \mathbf{W} & \\
 & & \mathrm{W} & \mathrm{L} & \mathbf{W} \\
 & & & \mathrm{W} & \mathrm{L}
\end{array}
\qquad
\begin{array}{cccccc}
C_1 & C_2 & C_3 & C_4 & C_5 & C_6 \\
\hline
\mathrm{W} & \mathrm{L} & \mathbf{W} & & & \\
 & \mathrm{W} & \mathrm{L} & \mathbf{W} & & \\
 & & \mathrm{W} & \mathrm{L} & \mathbf{W} & \\
 & & & \mathrm{W} & \mathrm{L} & \mathbf{W} \\
 & & & & \mathrm{W} & \mathrm{L}
\end{array}
$$

I plot in (27) the average number of errors made by the EDRA over ten runs on P-matrices corresponding to 5, 10, 15, 20 and 25 constraints. I consider four choices of the calibration constant, namely 0, 0.5, 0.7 and 0.9. See Appendix D for simulation results.

(27)



The plot shows that the number of errors made by the EDRA on P-matrices increases with the calibration constant and thus with the amount of constraint promotion performed: the number of errors is smallest for a null and a small calibration constant equal to 0.5; the number of errors is larger for a calibration constant equal to 0.7; it is largest for a calibration constant which is equal to 0.9 and thus approaches the forbidden threshold of 1. These simulation results show that the error-bound for the promotion/demotion learner cannot but be worse than the error-bound for the demotion-only learner in the general case.

These simulation results (27) can be interpreted as follows. Consider the ERC matrices obtained from the P-matrices by stripping the rightmost layer of w's, boldfaced in (26). Riggle [29] calls the resulting matrices *diagonal* (*D-matrices*). To illustrate, the D-matrices corresponding to $n = 4, 5, 6$ constraints are listed in (28).

(28)
$$
\begin{array}{cccc}
C_1 & C_2 & C_3 & C_4 \\
\hline
\mathrm{W} & \mathrm{L} & & \\
 & \mathrm{W} & \mathrm{L} & \\
 & & \mathrm{W} & \mathrm{L}
\end{array}
\qquad
\begin{array}{ccccc}
C_1 & C_2 & C_3 & C_4 & C_5 \\
\hline
\mathrm{W} & \mathrm{L} & & & \\
 & \mathrm{W} & \mathrm{L} & & \\
 & & \mathrm{W} & \mathrm{L} & \\
 & & & \mathrm{W} & \mathrm{L}
\end{array}
\qquad
\begin{array}{cccccc}
C_1 & C_2 & C_3 & C_4 & C_5 & C_6 \\
\hline
\mathrm{W} & \mathrm{L} & & & & \\
 & \mathrm{W} & \mathrm{L} & & & \\
 & & \mathrm{W} & \mathrm{L} & & \\
 & & & \mathrm{W} & \mathrm{L} & \\
 & & & & \mathrm{W} & \mathrm{L}
\end{array}
$$

The D-matrix with n constraints is the sparsest matrix (namely, the one with the largest number of blank entries) which requires the ranking $C_1 \gg C_2 \gg \ldots \gg C_n$. The D- and P-matrices differ because the latter has an additional layer of w's. These additional w's are irrelevant, as they do not contribute anything to consistency: the P-matrix is only consistent with the ranking $C_1 \gg C_2 \gg \ldots \gg C_n$, just as the D-matrix is. A demotion-only EDRA converges fast because it performs no constraint promotion and it is, therefore, insensitive to these additional irrelevant W's. An EDRA that performs both promotion and demotion converges more slowly because it is sensitive to the additional layer of irrelevant W's and it is, therefore, fouled around by them. Because of cases such as the P-matrix which have lots of irrelevant and distracting W's, error bounds for an EDRA which performs promotion as well as demotion cannot but be larger (namely worse) than the bounds for the demotion-only EDRA.

Yet, what about cases where all the W's are relevant to consistency? Does an EDRA which performs constraint promotion have an advantage in this case over the EDRA which only performs constraint demotion? To address this question, consider the *anti-Pater* ERC matrices (*antiP-matrices*) obtained from the D-matrices by adding a layer of additional W's at the left of the W's, rather than at the right of the L's as in the P-matrices. To illustrate, antiP-matrices corresponding to $n=4,5,6$ constraints are listed in (29), with the additional W's boldfaced.

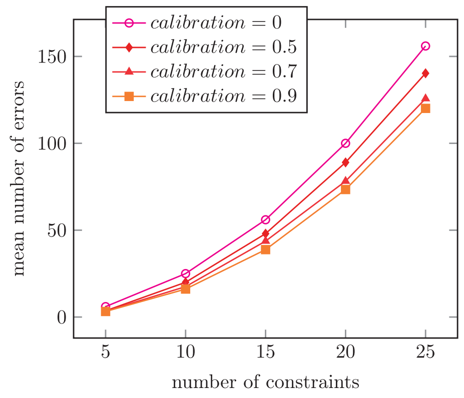
(29)
$$
\begin{array}{c}
\begin{array}{cccc} C_1 & C_2 & C_3 & C_4 \end{array} \\
\left[\begin{array}{cccc} \mathrm{W} & \mathrm{L} & & \\ \mathbf{W} & \mathrm{W} & \mathrm{L} & \\ & \mathbf{W} & \mathrm{W} & \mathrm{L} \end{array}\right]
\end{array}
\quad
\begin{array}{c}
\begin{array}{ccccc} C_1 & C_2 & C_3 & C_4 & C_5 \end{array} \\
\left[\begin{array}{ccccc} \mathrm{W} & \mathrm{L} & & & \\ \mathbf{W} & \mathrm{W} & \mathrm{L} & & \\ & \mathbf{W} & \mathrm{W} & \mathrm{L} & \\ & & \mathbf{W} & \mathrm{W} & \mathrm{L} \end{array}\right]
\end{array}
\quad
\begin{array}{c}
\begin{array}{cccccc} C_1 & C_2 & C_3 & C_4 & C_5 & C_6 \end{array} \\
\left[\begin{array}{cccccc} \mathrm{W} & \mathrm{L} & & & & \\ \mathbf{W} & \mathrm{W} & \mathrm{L} & & & \\ & \mathbf{W} & \mathrm{W} & \mathrm{L} & & \\ & & \mathbf{W} & \mathrm{W} & \mathrm{L} & \\ & & & \mathbf{W} & \mathrm{W} & \mathrm{L} \end{array}\right]
\end{array}
$$

The additional W's of antiP-matrices crucially contribute to consistency (contrary to the additional W's of P-matrices). In fact, D-matrices (as well as P-matrices) are only consistent with the ranking $C_1 \gg C_2 \gg \ldots \gg C_n$. While the additional W's ensure that antiP-matrices are consistent with any ranking which satisfies the ranking conditions (30), which subsume $C_1 \gg C_2 \gg \ldots \gg C_n$ as a special case (I am assuming for concreteness that the number n of constraints is odd). Consider, for instance, constraint C_3. In the case of the D-matrix and the P-matrix, constraint C_3 needs to be ranked underneath constraint C_2 which is in turn ranked underneath C_1. In the case of the antiP-matrix instead, it suffices to rank C_3 underneath C_1 because of the additional leftmost W corresponding to the second row of the antiP-matrix.

(30)
$$\begin{tikzcd}
 & C_1 \arrow[dl, dash] \arrow[dr, dash] & \\
C_2 \arrow[d, dash] & & C_3 \arrow[d, dash] \\
C_4 \arrow[d, dash, dotted] & & C_5 \arrow[d, dash, dotted] \\
C_{2k} \arrow[d, dash, dotted] & & C_{2k+1} \arrow[d, dash, dotted] \\
C_{n-1} & & C_n
\end{tikzcd}$$

I plot in (31) the average number of errors made by the EDRA over 10 runs on antiP-matrices corresponding to 5, 10, 15, 20 and 25 constraints. I consider four choices of the calibration constant, namely 0, 0.5, 0.7 and 0.9. See Appendix D for simulation results.

(31)



The plot shows that the number of errors made by the EDRA on antiP-matrices is largest when the calibration constant is equal to zero and the EDRA thus performs no constraint promotion; and it decreases as the calibration constant increases and the EDRA thus performs more and more constraint promotion. The simulation results (31) obtained for antiP-matrices are thus opposite to the simulation results (27) obtained for P-matrices.

Let me take stock. This section has started by looking at the case where lots of winner-preferring constraints are 'irrelevant' because they do not contribute to consistency, illustrated by P-matrices. The simulation results (27) on P-matrices show that constraint promotion is detrimental in this case, because the promotion component makes the EDRA sensitive to the irrelevant information. These simulation results make sense of the fact that the available error bounds get worse as the calibration constant increases and the EDRA thus performs more constraint promotion. This section has then proceeded to look at the complementary case where the winner-preferring constraints are all relevant as they all contribute to consistency, illustrated by antiP-matrices. The simulation results (31) on antiP-matrices show that constraint promotion affords a speed-up in this case. These simulation results thus suggest that one might be able to derive an error bound *specialized* to this case which is better for the promotion/demotion than for the demotion-only EDRA, contrary to the *general* error bound. At the current stage, I am not able to provide a specialized error-bound for antiP-matrices with such a property. I can nonetheless provide a result slightly weaker than that: fact 1 provides an error-bound (32) for the promotion-demotion EDRA on antiP-matrices which is, if not better, at least as good as the bound for the demotion-only learner. The simulation results in (31) with the calibration constant equal to zero show that this bound (32) is tight (namely cannot be improved) for the demotion only-case.

Fact 1

Express the promotion amount in terms of the number of winner-preferring constraints through a calibration constant, as in (11). Assume that the calibration constant is strictly smaller than 1. The number of updates made by this EDRA when trained on the antiP-matrix (29) can be bounded as follows:

(32) $$\text{Number of updates} \leq \frac{1}{4}(n-1)(n+1)$$

where n is the total number of constraints. ■

Appendix E provides a proof of this fact through a new line of analysis which bounds directly the numbers of updates triggered by each training triplet, whose sum yields the total number of errors.

6 Conclusions

A crucial architectural question in constraint-based phonology concerns the choice between the OT and HG modes of constraint interaction. At first sight, this might look like a typological issue: the choice between HG and OT should depend on their match with the actual typology of (attested and allegedly possible) natural language phonologies. Unfortunately, the problem of determining the proper mode of constraint interaction is unlikely to be settled on a purely typological basis. Let me make the nature of the difficulty explicit with a concrete example. Levelt *et al.* [19] report acquisition data where Dutch learning children go through stages where they allow for single marked structures while doubly marked structures lag behind. For instance, they allow syllables with complex codas and allow syllables with complex onsets but do not allow syllables with both complex edges. Do these intermediate acquisition stages provide evidence for the claim that child language displays gang-up effects and, therefore, requires HG over OT? The answer to this question is delicate. In fact, although *standard* OT does not capture these gang-up effects (with standard constraints for syllable types), Jäger and Rosenbach [16] show that the addition of a stochastic component allows OT to model gang-up effects in special configurations. Building on this observation, Jarosz [13] indeed shows that stochastic OT is able to model the gang-up effects reported in Levelt's child data. How can we adjudicate between the HG and the stochastic OT accounts of these gang-up effects?

Since *descriptive adequacy* is not likely to adjudicate between the OT and HG implementations of constraint-based phonology, various researchers have complemented descriptive with *explanatory adequacy*. The idea is to evaluate the two frameworks apart from their typological predictions, by distilling their algorithmic implications for modelling the production perception and acquisition of phonology. For instance, Riggle [29] and Bane *et al.* [1] compare the two frameworks from the perspective of learning-theoretic complexity measures such as their *VC-dimension*; Magri [23] compares them from the perspective of the computational problem of finding a *consistent* grammar; Jesney and Tessier [15] compare them from the perspective of the problem of finding a *restrictive* grammar. This article contributes to this enterprise from the perspective of the theory of error-driven learning.

The HG error-driven learner employed in the current literature (based on the Perceptron reweighing rule or its truncated variant; [25]) does not tolerate the stochastic implementation. In fact, Magri [24] constructs a counterexample with just 10 constraints where the HG stochastic learner makes over 1 million additional errors. Furthermore, the HG learner is not robust to noise. In fact, [24] constructs a counterexample with just 10 constraints where a single faulty update by a noisy piece of data requires the HG learner to perform over 30,000 additional updates to recover. On those same test cases, the stochastic implementation causes the OT learner to make less than 150 additional errors. Furthermore, the single noisy piece of data triggers only nine additional updates by the OT learner. This sharp contrast between the HG and the OT learners holds despite the fact that the violation profiles in these counterexamples have been chosen in such a way that the HG and OT typologies explored by the two learners exactly coincide.

The present article has strengthened this contrast between HG and OT error-driven learning by supporting the OT simulation results with analytical error bounds. The analysis of the stochastic OT learner presented in Section 3 guarantees that the number of additional errors made by the OT learner because of the stochastic implementation grows slowly with the number of constraints. Analogously, the analysis of the OT learner in the noisy setting presented in Section 4 guarantees that the number of additional updates needed by the OT learner to recover from corrupted data grows slowly with the number of constraints. These results thus make sense of the fact that the counterexamples in [24] do not foul the OT learner. And it ensures that no analogous counterexample can never be construed against the OT learner.

To illustrate the implications of these results, let me go back to the problem of modelling Levelt's data on the acquisition of Dutch syllable types. As explained above, these acquisition data display lags in the production of doubly marked structures that can be modelled through both HG and (stochastic) OT. The learnability perspective adopted in this article might then help to adjudicate between those two models of the acquisition data. In fact, while the HG learner is not efficient from a constraint-independent perspective, the stochastic implementation does not affect the efficiency of the OT learner, as guaranteed by Theorem 2 in Section 3 above. These computational results might thus provide a starting point for a learnability-based argument that fills the gap left by the insufficiency of sheer phonological data.

Acknowledgements

The research reported in this article has been supported by a Marie Curie Intra European Fellowship (Grant agreement number: PIEF-GA-2011-301938).

References

[1] M. Bane, J. Riggle and M. Sonderegger. The VC dimension of constraint-based grammars. *Lingua*, **120**, 1194–1208, 2010.

[2] T. Biró. *Finding the Right Words: implementing optimality theory with simulated annealing*. Ph.D. thesis, University of Groningen, GroDiL 62. ROA-896, 2006.

[3] Boersma P. How we learn variation, optionality and probability. In R. van Son, ed., *Proceedings of the Institute of Phonetic Sciences (IFA) 21*, pp. 43–58, University of Amsterdam: Institute of Phonetic Sciences, 1997.

[4] P. Boersma. *Functional Phonology*. Ph.D. thesis, University of Amsterdam, The Netherlands. The Hague: Holland Academic Graphics, 1998.

[5] P. Boersma. Some correct error-driven versions of the constraint demotion algorithm. *Linguistic Inquiry*, **40**, 667–686, 2009.

[6] P. Boersma and B. Hayes. Empirical tests for the Gradual Learning Algorithm. *Linguistic Inquiry*, **32**, 45–86, 2001.

[7] P. Boersma and J. Pater. Convergence properties of a gradual learning algorithm for Harmonic Grammar. In J. McCarthy and J. Pater, eds, *Harmonic Grammar and Harmonic Serialism*. Equinox Press, to appear.

[8] A. W. Coetzee and S. Kawahara. Frequency biases in phonological variation. *Natural Language and Linguistic Theory*, **31**, 47–89, 2013.

[9] A. W. Coetzee and J. Pater. Weighted constraints and gradient restrictions on place co-occurrence in Muna and Arabic. *Natural Language and Linguistic Theory*, **26**, 289–337, 2008.

[10] A. W. Coetzee and J. Pater. The place of variation in phonological theory. In J. Goldsmith, J. Riggle, and A. Yu, eds, *Handbook of phonological theory*, pp. 401–434. Blackwell, 2011.

[11] R. Frank and S. Kapur. On the use of triggers in parameter setting. *Linguistic Inquiry*, **27**, 623–660, 1996.

[12] E. Gibson and K. Wexler. Triggers. *Linguistic Inquiry*, **25**, 407–454, 1994.

[13] G. Jarosz. Implicational markedness and frequency in constraint-based computational models of phonological learning. *Journal of Child Language*, **37**, 565–606, 2010.

[14] G. Jarosz. Learning with hidden structure in Oprimality Theory and Harmonic Grammar: beyond robust interpretative parsing. *Phonology*, **30**, 27–71, 2013.

[15] K. Jesney and A.-M. Tessier. Biases in Harmonic Grammar: the road to restrictive learning. *Natural Language and Linguistic Theory*, **29**, 251–290, 2011.
[16] G. Jäger and A. Rosenbach. The winner takes it all–almost. Cumulativity in grammatical variation. *Linguistics*, **44**, 937–971, 2006.
[17] G. Legendre, Y. Miyata and P. Smolensky. Harmonic grammar: a formal multi-level connectionist theory of linguistic well-formedness: an application. In M. A. Gernsbacher and S. J. Derry, eds, *Annual Conference of the Cognitive Science Society 12*, pp. 884–891, Lawrence Erlbaum Associates, 1998b.
[18] G. Legendre, Y. Miyata and P. Smolensky. Harmonic grammar: a formal multi-level connectionist theory of linguistic well-formedness: theoretical foundations. In M. A. Gernsbacher and S. J. Derry, eds, *Annual Conference of the Cognitive Science Society 12*, pp. 388–395, Lawrence Erlbaum, 1998a.
[19] C. C. Levelt, N. O. Schiller and W. J. Levelt. The acquisition of syllable types. *Language Acquisition*, **8(3)**, 237–264, 2000.
[20] G. Magri. Constraint promotion: not only convergent, but also efficient. In *Proceedings of the 48th annual conference of the Chicago Linguistics Society*, 2012.
[21] G. Magri. Convergence of error-driven ranking algorithms. *Phonology*, **29**, 213–269, 2012.
[22] G. Magri. The complexity of learning in OT and its implications for the acquisition of phonotactics. *Linguistic Inquiry*, **44**, 433–468, 2013.
[23] G. Magri. HG has no computational advantages over OT: towards a new toolkit for computational OT. *Linguistic Inquiry*, **44**, 569–609, 2013.
[24] G. Magri. Error-driven learning in OT and HG: a comparison. Manuscript, 2014.
[25] G. Magri. How to keep the HG weights non-negative: the truncated perceptron reweighing rule. Manuscript, 2014.
[26] J. Pater. Gradual learning and convergence. *Linguistic Inquiry*, **39**, 334–345, 2008.
[27] A. Prince. Entailed ranking arguments. Ms., Rutgers University, Rutgers Optimality Archive, ROA 500, 2002. Available at http://www.roa.rutgers.edu.
[28] A. Prince and P. Smolensky. *Optimality Theory: Constraint Interaction in Generative Grammar*. Blackwell, 2004. As Technical Report CU-CS-696-93, Department of Computer Science, University of Colorado at Boulder, and Technical Report TR-2, Rutgers Center for Cognitive Science, Rutgers University, New Brunswick, NJ, April 1993. Also available as ROA 537 version.
[29] J. Riggle. The complexity of ranking hypotheses in Optimality Theory. *Computational Linguistics*, **35**, 47–59, 2009.
[30] P. Smolensky and G. Legendre. *The Harmonic Mind*. MIT Press, 2006.
[31] B. Tesar and P. Smolensky. Learnability in Optimality Theory. *Linguistic Inquiry*, **29**, 229–268, 1998.

Received 10 December 2014

Appendix A. Error-bound for the deterministic EDRA in the noise-free setting

This section reviews Tesar and Smolensky's [31] proof of Lemma 1 for the deterministic EDRA in the noise-free setting, repeated below. The proof formalizes the following intuition: since the re-ranking rule (3) only demotes the constraints that need to be demoted (namely, those that are currently undominated), then no constraint can be demoted too low.

Lemma 1
Suppose that an EDRA is trained on a sequence of comparison triplets consistent with some constraint ranking. Without loss of generality, assume that ranking is $C_1 \gg C_2 \gg \ldots \gg C_k \gg \ldots$, whereby constraint C_k is assigned to the k-th slot (the first slot being the top of the ranking). The current ranking value θ_k^t of constraint C_k entertained by the EDRA at an arbitrary time t can be bounded as follows:

(A.1) $\theta_k^t \geq -(k-1)$

so that the sum of the current ranking values can be bounded as follows:

(A.2) $$\sum_{k=1}^{n} \theta_k^t \geq -\sum_{k=1}^{n}(k-1) = -\frac{1}{2}n(n-1)$$

where n is the total number of constraints. ■

Proof. The proof of (A.1) is by induction on k. As the basis of induction, let me establish the bound for $k=1$, which boils down to the inequality (A.3) for the ranking value θ_1^t of constraint C_1 at an arbitrary time t. The training triplets are by assumption consistent with a ranking that assigns C_1 to the top. This means that constraint C_1 cannot be loser-preferring according to any training triplet and is, therefore, never demoted. Its ranking value θ_1^t thus starts null and never decreases, ensuring (A.3).

(A.3) $\theta_1^t \geq 0$

As the inductive hypothesis, assume that the bound (A.1) holds up to $k-1$. Thus in particular, the ranking values $\theta_1^t, \ldots, \theta_{k-1}^t$ of constraints $C_1, \ldots, C_{k-1}$ at any time t satisfy (A.4).

(A.4) $\theta_1^t, \theta_2^t, \ldots, \theta_{k-1}^t \geq -((k-1)-1) = -k+2$

As the inductive step, let me show that the bound (A.1) holds for k as well. By contradiction, assume that it does not. This means that there exists a time t such that constraint C_k is demoted between times t and $t+1$ and its current ranking value drops below the forbidden threshold at time $t+1$, as stated in (A.5).

(A.5) $\theta_k^{t+1} < -(k-1)$

In order for C_k to have been demoted in between times t and $t+1$, it must have been a loser-preferring constraint relative to the comparative triplet which has triggered that update. Since that triplet is by hypothesis consistent with the ranking $C_1 \gg C_2 \gg \ldots C_{k-1} \gg C_k \gg \ldots$, at least one of the constraints $C_1, C_2, \ldots, C_{k-1}$ ranked above C_k must be winner-preferring relative to that triplet. Let C_h with $h \in \{1, \ldots, k-1\}$ be a winner-preferring constraint. In order for the loser-preferrer C_k to have been demoted in between times t and $t+1$, it must have been undominated. This means in particular that the ranking value θ_k^t of the loser-preferrer C_k at time t must have been at least as large as the ranking value θ_h^t of the winner-preferring constraint C_h at time t, as stated in (A.6).

(A.6) $\theta_k^t \geq \theta_h^t$

Since the constraint C_k has been demoted by 1 in between times t and $t+1$, its ranking value θ_k^t at time t before the update is larger by 1 than its ranking value θ_k^{t+1} after the update, as stated in (A.7a). In (A.7b), I have used the contradictory assumption (A.5). In (A.7c), I have used the inductive hypothesis (A.4), as $h \in \{1, \ldots, k-1\}$ by hypothesis.

(A.7)
$$\begin{aligned}\theta_k^t &\overset{(a)}{=} \theta_k^{t+1}+1\\ &\overset{(b)}{<} -(k-1)+1\\ &= -k+2\\ &\overset{(c)}{\leq} \theta_h^t\end{aligned}$$

The two inequalities (A.6) and (A.7) contradict each other, completing the proof of the bound (A.1) on each individual current ranking value. The bound (A.2) on the sum of the current ranking values follows from (A.1) together with the identity $\sum_{k=1}^{n}(k-1)=\frac{1}{2}n(n-1)$. ■

Appendix B. Error-bound for the stochastic EDRA in the noise-free setting

This section presents a proof of Lemma 2 for the stochastic EDRA in the noise-free setting, repeated below. The proof is analogous to Tesar and Smolensky's [31] proof of Lemma 1. It formalizes the following intuition: since the re-ranking rule (3) only demotes the constraints that need to be demoted (namely, those that are currently undominated) and since the stochastic ranking values $\theta_k+\epsilon_k$ cannot be too different from the current ranking values θ_k (because the stochastic values ϵ_k are bounded between $-\Delta$ and $+\Delta$), no constraint can be demoted too low.

LEMMA 2
Suppose that a stochastic EDRA is trained on a sequence of comparative triplets consistent with some constraint ranking. Without loss of generality, assume that ranking is $C_1 \gg C_2 \gg \ldots \gg C_k \gg \ldots$, whereby constraint C_k is assigned to the k-th slot (the first slot being the top of the ranking). Suppose furthermore that the stochastic values ϵ_k are always bounded between $-\Delta$ and $+\Delta$, for some threshold $\Delta \geq 0$. The current ranking value θ_k^t of constraint C_k entertained by the stochastic EDRA at an arbitrary time t can be bounded as follows:

(A.8) $\theta_k^t \geq -(k-1)-2\Delta(k-1)$

so that the sum of the current ranking values can be bounded as follows:

(A.9)
$$\sum_{k=1}^{n}\theta_k^t \geq -\sum_{k=1}^{n}(k-1)-2\Delta\sum_{k=1}^{n}(k-1)=-\frac{1}{2}n(n-1)-\Delta n(n-1)$$

where n is the total number of constraints. ■

PROOF. The proof of (A.8) is by induction on k. As the basis of induction, I note that the bound (A.8) holds for $k=1$ because it boils down to the inequality (A.3), which holds in the stochastic case for exactly the same reason it holds in the deterministic case. As the inductive hypothesis, assume that the bound (A.8) holds up to $k-1$. Thus in particular, the ranking values $\theta_1^t,\ldots,\theta_{k-1}^t$ of constraints $C_1,\ldots,C_{k-1}$ at any time t satisfy the inequality (A.10).

(A.10) $\theta_1^t,\theta_2^t,\ldots,\theta_{k-1}^t \geq -((k-1)-1)-2((k-1)-1)\Delta = -(k-2)-2(k-2)\Delta$

As the inductive step, let me show that the bound (A.8) holds for k as well. By contradiction, assume that it does not. This means that there exists a time t such that constraint C_k is demoted between times t and $t+1$ and its current ranking value drops below the forbidden threshold at time $t+1$, as stated in (A.11).

(A.11) $\theta_k^{t+1} < -(k-1)-2(k-1)\Delta$

In order for C_k to have been demoted in between times t and $t+1$, it must have been a loser-preferring constraint relative to the comparative triplet which has triggered that update. Since that triplet is by hypothesis consistent with the ranking $C_1 \gg C_2 \gg \ldots C_{k-1} \gg C_k \gg \ldots$, at least one of the constraints $C_1, C_2, \ldots, C_{k-1}$ ranked above C_k must be winner-preferring relative to that triplet. Let C_h with $h \in \{1, \ldots, k-1\}$ be a winner-preferring constraint. In order for the loser-preferrer C_k to have been demoted in between times t and $t+1$, it must have satisfied condition (14) for stochastic undominatedness. Thus, the stochastic ranking value $\theta_k^t + \epsilon_k^t$ of the loser-preferrer C_k at time t must in particular have been at least as large as the stochastic ranking value $\theta_h^t + \epsilon_h^t$ of the winner-preferrer C_h at time t, as stated in (A.6).

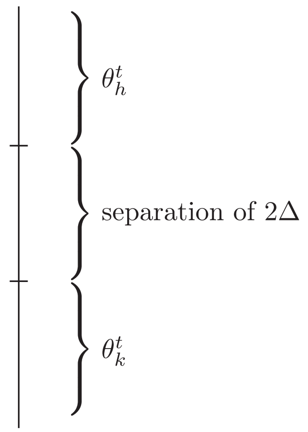
(A.12) $\theta_k^t + \epsilon_k^t \geq \theta_h^t + \epsilon_h^t$

Since constraint C_k has been demoted by 1 in the update between times t and $t+1$, its ranking value θ_k^t at time t before the update is larger by 1 than its ranking value θ_k^{t+1} after the update, as stated in (A.7a). In (A.13b), I have then used the contradictory assumption (A.11). In (A.13c), I have used the inductive hypothesis (A.10), as $h \in \{1, \ldots, k-1\}$ by hypothesis.

(A.13)
$$\begin{aligned} \theta_k^t &\overset{(a)}{=} \theta_k^{t+1} + 1 \\ &\overset{(b)}{<} -(k-1) - 2(k-1)\Delta + 1 \\ &= -(k-2) - 2(k-2)\Delta - 2\Delta \\ &\overset{(c)}{\leq} \theta_h^t - 2\Delta \end{aligned}$$

The chain of inequalities (A.13) says that the ranking value θ_k^t of constraint C_k at time t is smaller than the ranking value θ_h^t of constraint C_h at that time t by more than 2Δ, as represented in (A.14).

(A.14)



Since the stochastic values ϵ_h^t and ϵ_k^t corresponding to constraints C_h and C_k are bounded between $-\Delta$ and $+\Delta$, (A.14) says that the stochastic ranking value $\theta_k^t + \epsilon_k$ of constraint C_k at time t is strictly smaller than the stochastic ranking value $\theta_h^t + \epsilon_h^t$ of the constraint C_h at that time t, as stated in (A.15).

(A.15) $\theta_k^t + \epsilon_k^t < \theta_h^t + \epsilon_h^t$

The two inequalities (A.12) and (A.15) contradict each other, completing the proof of the bound (A.8) on each individual current ranking value. The bound (A.9) on the sum of the current ranking values follows from (A.8) together with the identity $\sum_{k=1}^{n}(k-1) = \frac{1}{2}n(n-1)$. ■

Appendix C. Error bound for the deterministic EDRA in the noisy setting

This section presents a proof of Lemma 3 for the deterministic EDRA in the noisy setting, repeated below. The proof is analogous to Tesar and Smolensky's [31] proof of Lemma 1. It formalizes the following intuition: since the re-ranking rule (3) only demotes the constraints that need to be demoted (namely, those that are currently undominated) and since the corrupted training triplets are not too many (as quantified by the numbers $\delta_1, \delta_2, \ldots$), no constraint can be demoted too low.

LEMMA 3
Suppose that the EDRA's training sequence consists of both *pristine* and *corrupted* training triplets. The pristine triplets are all consistent with some constraint ranking. Without loss of generality, assume that ranking is $C_1 \gg C_2 \gg \ldots \gg C_k \gg \ldots$, whereby constraint C_k is assigned to the k-th slot (the first slot being the top of the ranking). The corrupted triplets are instead inconsistent with this ranking $C_1 \gg C_2 \gg \ldots \gg C_k \gg \ldots$ and thus count as noise. Assume that the training sequence contains only a finite number of these corrupted triplets (while the pristine triplets can be infinite in number), and let δ_h be the number of corrupted triplets in the training sequence where constraint C_h is loser-preferring. The current ranking value θ_k^t of constraint C_k entertained by the deterministic EDRA at an arbitrary time t can be bounded as follows:

(A.16) $$\theta_k^t \geq -(k-1) - \sum_{h=1}^{k} \delta_h$$

so that the sum of the current ranking values can be bounded as follows:

(A.17) $$\theta_k^t \geq -\sum_{k=1}^{n}(k-1) - \sum_{k=1}^{n}\sum_{h=1}^{k} \delta_h \geq -\frac{1}{2}n(n-1) - n(n-1)\delta$$

where n is the number of constraints and δ is the number of corrupted triplets in the training sequence. ■

PROOF. The proof of (A.16) is by induction on k. As the basis of induction, let me establish the bound for $k=1$, which boils down to the inequality (A.18) for the ranking value θ_1^t of constraint C_1 at an arbitrary time t. The pristine training triplets are by assumption consistent with a ranking which assigns C_1 to the top. This means that constraint C_1 cannot be loser-preferring according to any pristine training triplet and is, therefore, never demoted by them. Constraint C_1 can thus only be demoted by the corrupted comparative triplets where it is loser-preferring. By assumption, the training sequence contains only δ_1 of them. Since constraint C_1 starts with a null initial ranking value and is decreased by 1 for at most δ_1 times, its ranking value θ_1^t can never drop below $-\delta_1$, ensuring (A.18).

(A.18) $$\theta_1^t \geq -\delta_1$$

As the inductive hypothesis, assume that the bound (A.16) holds up to $k-1$. This entails in particular that the ranking values $\theta_1^t, \ldots, \theta_{k-1}^t$ of the constraints $C_1, \ldots, C_{k-1}$ at any time t can all be bounded as in (A.19).

(A.19) $$\theta_1^t, \ldots, \theta_{k-1}^t \geq -((k-1)-1) - \sum_{h=1}^{k-1} \delta_h = -(k-2) - \sum_{h=1}^{k-1} \delta_h$$

As the inductive step, let me show that the bound (A.16) then holds for k at any time t. The proof of this latter claim is in turn by induction on δ_k. As the basis of the induction on δ_k, let me show that the bound (A.16) is true for k at any time t when $\delta_k = 0$, namely when the training sequence contains no corrupted triplets where constraint C_k is loser-preferring. By contradiction, assume that the bound does not hold in this case. This means that there exists a time t such that the constraint C_k is demoted between times t and $t+1$ and its current ranking value drops below the forbidden threshold at time $t+1$, as stated in (A.20), whose left-hand side is indeed (A.16) with $\delta_k = 0$.

$$\text{(A.20)} \quad \theta_k^{t+1} < -(k-1) - \sum_{h=1}^{k-1} \delta_h$$

In order for C_k to have been demoted in between times t and $t+1$, it must have been a loser-preferring constraint relative to the comparative triplet which has triggered that update. The hypothesis $\delta_k = 0$ says that the training sequence contains no corrupted triplets where constraint C_k is loser-preferring. Thus, the update in between times t and $t+1$ must have been triggered by a pristine triplet. Reasoning as in the proof of the two preceding Lemmas, I conclude that there exists a constraint C_h with $h \in \{1, \ldots, k-1\}$ such that (A.21) holds.

$$\text{(A.21)} \quad \theta_k^t \geq \theta_h^t$$

Since constraint C_k has been demoted by 1 in the update between times t and $t+1$, its ranking value θ_k^t of C_k at time t before the update is larger by 1 than its ranking value θ_k^{t+1} after the update, as stated in (A.22a). In (A.22b), I have used the contradictory assumption (A.20). In step (A.22c), I have used the inductive hypothesis (A.19), as $h \in \{1, \ldots, k-1\}$ by hypothesis. The two inequalities (A.21) and (A.22) contradict each other.

$$\text{(A.22)} \quad \begin{aligned} \theta_k^t &\overset{(a)}{=} \theta_k^{t+1} + 1 \\ &\overset{(b)}{<} -\sum_{h=1}^{k-1} \delta_h - (k-1) + 1 \\ &= -\sum_{h=1}^{k-1} \delta_h - (k-2) \\ &\overset{(c)}{\leq} \theta_h^t \end{aligned}$$

As the hypothesis of the induction on δ_k, assume now that the bound (A.16) holds when the training sequence contains $\delta_k - 1$ corrupted triplets where constraint C_k is loser-preferring. In this case, the bound takes the shape (A.23).

$$\text{(A.23)} \quad \theta_k^t \geq -(k-1) - \sum_{h=1}^{k-1} \delta_h - (\delta_k - 1) = -(k-2) - \sum_{h=1}^{k} \delta_h$$

As the inductive step, let me show that the bound holds for δ_k. By contradiction, assume that the bound does not hold. This means that there exists a time t such that constraint C_k is demoted between times t and $t+1$ and its current ranking value drops below the forbidden threshold at time $t+1$, as stated in (A.24).

$$\text{(A.24)} \quad \theta_k^{t+1} < -\sum_{h=1}^{k} \delta_h - (k-1)$$

The update between times t and $t+1$ which has demoted constraint C_k can in principle have been triggered by either a corrupted or a pristine triplet. To start, suppose that it has been triggered by a corrupted triplet. By hypothesis, the training sequence contains a total of δ_k corrupted triplets where constraint C_k is loser-preferring. Since one such triplet triggers an update between times t and $t+1$, the training sequence up to time t can contain at most $\delta_k - 1$ corrupted triplets which demote C_k. The inductive hypothesis (A.23) thus applies, contradicting (A.24). Thus, the update in between times t and $t+1$ which has demoted constraint C_k must have been triggered by one of the pristine triplets which are consistent with the ranking $C_1 \gg C_2 \gg \ldots \gg C_{k-1} \gg C_k \gg \ldots$. By reasoning as in the two preceding Lemmas, I conclude that there exists a constraint C_h with $h \in \{1, \ldots, k-1\}$ such that (A.25) holds.

$$\text{(A.25)} \quad \theta_k^t \geq \theta_h^t$$

Since constraint C_k has been demoted by 1 in the update between times t and $t+1$ the current ranking value θ_k^t of C_k at time t before the update is larger by 1 than its ranking value θ_k^{t+1} after the update, as stated in (A.26a). In (A.26b), I have used the contradictory assumption (A.24). In (A.26c), I have used the inductive hypothesis (A.19), as $h \in \{1, \ldots, k-1\}$ by hypothesis.

$$\text{(A.26)} \quad \begin{aligned} \theta_k^t &\overset{(a)}{=} \theta_k^{t+1} + 1 \\ &\overset{(b)}{<} -\sum_{h=1}^{k} \delta_h - (k-1) + 1 \\ &= -\sum_{h=1}^{k} \delta_h - (k-2) \\ &\overset{(c)}{\leq} \theta_h^t \end{aligned}$$

The two inequalities (A.25) and (A.26) contradict each other. This concludes the inductive proof of the bound (A.16) on each individual current ranking value. The bound (A.17) on the sum of the current ranking values follows straightforwardly from (A.16) together with the fact that $\delta_1 + \delta_2 + \ldots + \delta_n \leq \delta(n-1)$, because each comparative triplet can have at most $n-1$ loser-preferring constraints.[6] ■

Appendix D. Simulation results on P- and anti-P-matrices

Tables (A.27)-(A.28) report the total number of errors made by the EDRA with different values of the calibration constant (0, 0.5, 0.7 and 0.9) in 10 runs I-X on P-matrices (26) and on anti-P-matrices (29) corresponding to $n = 5, 10, 15, 20, 25$ constraints. The EDRA is trained on triplets described by the rows of the matrices, sampled at random with equal probability.

[6]A comparative triplet which has n loser-preferring constraints has no winner-preferring constraints and can, therefore, be immediately spotted by the EDRA as corrupted, thus avoiding performing the corresponding update.

(A.27) Simulation results on P-matrices:

n	cal.	I	II	III	IV	V	VI	VII	VIII	IX	X	Mean	SD
5	0	10	10	10	10	10	10	10	10	10	10	10.00	0.00
	0.5	11	11	11	11	11	11	11	11	11	11	11.00	0.00
	0.7	8	8	8	8	8	8	8	8	8	8	8.00	0.00
	0.9	8	8	8	8	8	8	8	8	8	8	8.00	0.00
10	0	45	45	45	45	45	45	45	45	45	45	45.00	0.00
	0.5	44	44	44	44	44	44	44	44	44	44	44.00	0.00
	0.7	76	76	76	76	76	76	76	76	76	76	76.00	0.00
	0.9	97	97	97	97	97	97	97	115	97	97	98.80	5.40
15	0	105	105	105	105	105	105	105	105	105	105	105.00	0.00
	0.5	102	102	102	102	102	102	102	102	102	102	102.00	0.00
	0.7	186	186	186	186	186	186	186	191	186	186	186.50	1.50
	0.9	367	367	367	367	367	367	367	367	367	367	367.00	0.00
20	0	190	190	190	190	190	190	190	190	190	190	190.00	0.00
	0.5	190	190	190	190	190	190	190	190	190	190	190.00	0.00
	0.7	353	353	353	353	353	353	353	353	353	353	353.00	0.00
	0.9	735	735	735	735	735	735	735	735	735	735	735.00	0.00
25	0	300	300	300	300	300	300	300	300	300	300	300.00	0.00
	0.5	300	300	300	300	300	300	300	300	300	300	300.00	0.00
	0.7	564	564	566	564	564	564	564	564	564	564	564.20	0.60
	0.9	1225	1225	1225	1225	1225	1225	1225	1225	1225	1225	1225.00	0.00

(A.28) Simulation results on antiP-matrices:

n	cal.	I	II	III	IV	V	VI	VII	VIII	IX	X	Mean	SD
5	0	6	6	6	6	6	6	6	6	6	6	6.00	0.00
	0.5	4	4	4	4	4	4	4	2	4	2	3.60	0.80
	0.7	4	4	4	2	4	2	2	4	4	4	3.40	0.92
	0.9	2	4	2	4	4	4	2	2	4	4	3.20	0.98
10	0	25	25	25	25	25	25	25	25	25	25	25.00	0.00
	0.5	20	20	20	20	20	20	20	20	20	20	20.00	0.00
	0.7	19	19	19	19	16	16	16	16	19	16	17.50	1.50
	0.9	19	15	15	15	16	16	15	16	19	15	16.10	1.51
15	0	56	56	56	56	56	56	56	56	56	56	56.00	0.00
	0.5	48	48	48	48	48	48	48	48	48	48	48.00	0.00
	0.7	45	45	43	45	43	45	45	45	45	36	43.70	2.69
	0.9	41	41	41	27	35	44	41	41	41	36	38.80	4.66
20	0	100	100	100	100	100	100	100	100	100	100	100.00	0.00
	0.5	89	89	89	89	89	89	89	89	89	89	89.00	0.00
	0.7	79	79	79	79	78	79	79	72	79	79	78.20	2.09
	0.9	71	77	70	77	69	76	71	76	76	71	73.40	3.07
25	0	156	156	156	156	156	156	156	156	156	156	156.00	0.00
	0.5	140	143	140	140	140	140	140	140	140	140	140.30	0.90
	0.7	133	128	128	128	128	128	119	127	119	119	125.70	4.65
	0.9	119	119	124	117	118	127	114	117	127	119	120.10	4.18

Appendix E. Error-bound on anti-P-matrices

Recall that the anti-P-matrix has the shape in (A.29). It has n columns corresponding to the constraints $C_1, \ldots, C_n$ and $n-1$ rows which describe in ERC notation the training comparative triplets. The k-th column corresponding to constraint C_k has only three entries which are not blank: the entry in the $(k-1)$-th row, which is equal to L; and the entries in the k-th and $(k+1)$-th rows, which are equal to W.

(A.29)

$$\begin{array}{r|ccccccccccccc|}
 & C_1 & C_2 & C_3 & C_4 & \ldots & & C_{k-1} & C_k & C_{k+1} & & \ldots & & C_n \\
\text{ERC}\,1 & \text{W} & \text{L} & & & & & & & & & & & \\
\text{ERC}\,2 & \text{W} & \text{W} & \text{L} & & & & & & & & & & \\
\text{ERC}\,3 & & \text{W} & \text{W} & \text{L} & & & & & & & & & \\
\text{ERC}\,4 & & & \text{W} & \text{W} & \text{L} & & & & & & & & \\
\vdots & & & & & \ddots & & & & & & & & \\
\text{ERC}\,k{-}2 & & & & & \text{W} & \text{W} & \text{L} & & & & & & \\
\text{ERC}\,k{-}1 & & & & & & \text{W} & \text{W} & \text{L} & & & & & \\
\text{ERC}\,k & & & & & & & \text{W} & \text{W} & \text{L} & & & & \\
\text{ERC}\,k{+}1 & & & & & & & & \text{W} & \text{W} & \text{L} & & & \\
\text{ERC}\,k{+}2 & & & & & & & & & \text{W} & \text{W} & \text{L} & & \\
\vdots & & & & & & & & & & & \ddots & & \\
\text{ERC}\,n{-}1 & & & & & & & & & & & \text{W} & \text{W} & \text{L}
\end{array}$$

This section presents a proof of Fact 1 repeated below, which provides the error bound (A.30) specialized to antiP-matrices. This bound is better (namely smaller) than the error bound (13) provided by Theorem 1 for the general case. Furthermore, it does not depend on the calibration constant and thus holds unchanged for both the demotion-only and the promotion/demotion EDRA.

FACT 1
Express the promotion amount in terms of the number of winner-preferring constraints through a calibration constant, as in (11). Assume that the calibration constant is strictly smaller than 1. The number of updates made by this EDRA when trained on the anti-P-matrix (A.29) can be bounded as follows:

(A.30) $$\text{Number of updates} \leq \frac{1}{4}(n-1)(n+1)$$

where n is the total number of constraints. ■

Before turning to the actual proof of Fact 1, let me repeat the reasoning outlined in Section 2 and detailed in Appendix A, slightly adapted to the special case of anti-P-matrices. Anti-P-matrices are consistent with the ranking (30). The ranking value θ_k for constraint C_k entertained by the EDRA at an arbitrary iteration when trained on the anti-P-matrix can then be lower-bounded as in (A.31), where I am assuming for concreteness that the number n of constraints is odd. For instance, constraint C_1 is never loser-preferring in the anti-P-matrix, and its ranking value θ_1 never drops below zero. Constraints C_2 and C_3 only need to be demoted underneath C_1, and thus their ranking values θ_2 and θ_3 can only drop to -1 but not any further. Constraints C_4 and C_5 only need to be demoted underneath C_2 and C_3, and thus their ranking values θ_4 and θ_5 can only drop to -2 but not any further. And so on.

(A.31)
$$\begin{aligned}\theta_1 &\geq 0\\ \theta_2, \theta_3 &\geq -1\\ \theta_4, \theta_5 &\geq -2\\ &\vdots\\ \theta_{2k}, \theta_{2k+1} &\geq -k\\ &\vdots\\ \theta_{n-1}, \theta_n &\geq -\frac{n-1}{2}\end{aligned}$$

By (A.31), the sum of the current ranking values is never smaller than $2\sum_{k=1}^{\frac{n-1}{2}}(-k) = -\frac{1}{4}(n-1)(n+1)$. Combining with the inequality (12), we obtain the error bound (A.32) for an EDRA which adopts a promotion amount of the form (11) with a calibration constant which is strictly smaller than 1.

(A.32)
$$\text{Number of errors} \leq \frac{1}{4}\frac{1}{1-\text{calibration}}(n-1)(n+1)$$

The bound (A.32) specialized for anti-P-matrices is better (namely smaller) than the bound provided by theorem 1 for the general case. If the EDRA performs constraint demotion only and thus adopts a calibration constant equal to zero, the bound (A.32) becomes (A.30). Unfortunately, the bound (A.32) depends on the calibration constant and gets worse (namely larger) as the calibration constant increases and the learner thus performs more and more constraint promotion. A different line of analysis of the promotion/demotion EDRA on antiP-matrices is thus needed in order to prove the calibration-independent error bound provided by Fact 1. The core intuition developed in the following proof is as follows. The analysis just illustrated bounds the number of errors indirectly, through a bound on the current ranking values. The alternative analysis pursued by the following proof instead bounds directly the numbers of updates triggered by each training triplet, whose sum is the total number of errors.

Proof. Let α_k^t be the number of updates triggered by the k-th ERC of the antiP-matrix (A.29) up to time t in the run considered. The proof has two parts: the first part establishes the bounds (A.33); the second part then derives the error bound (A.30).

(A.33)
a. $\alpha_1^t \leq 1$
b. $\alpha_2^t \leq 1$
c. $\alpha_k^t \leq \alpha_{k-2}^t + 1$ for $k = 3, 4, \ldots, n-1$

The proof of the bounds (A.33) is by induction on time t. As the basis of the induction, note that the bounds (A.33) hold at time $t = 0$, as in this case no update has been triggered yet and thus $\alpha_k^{t=0} = 0$ for every $k = 1, \ldots, n-1$. As the inductive hypothesis, assume that the bounds (A.33) hold at time t. As the inductive step, let me then show that the bounds (A.33) also hold at time $t+1$.

To start, let me prove that the bound (A.33a) holds at time $t+1$. If it is not ERC 1 that has triggered the update in between times t and $t+1$, then $\alpha_1^{t+1} = \alpha_1^t$, and the inductive hypothesis that the bound (A.33a) holds at time t immediately entails that it holds at time $t+1$ as well. Thus, assume that it is indeed ERC 1 that has triggered an update in between times t and $t+1$. The chain of implications in (A.34) thus holds. Whenever ERC 1 triggers an update, its loser-preferring constraint C_2 must be undominated, namely its ranking value must be at least as large as the ranking value of the winner-preferring constraint C_1, as stated in (A.34a). From now on, let $0 \leq c < 1$ be the calibration constant. Step (A.34b) follows by expressing the ranking values θ_1^t and θ_2^t of constraints C_1 and C_2 in terms of the number of updates triggered by the various ERCs. For instance, constraint C_1 is demoted by -1 by ERC 1 and promoted by $c/2$ by both ERCs 2 and 3. Its ranking value θ_1^t at time t is thus equal to $-\alpha_1^t + \frac{c}{2}\alpha_2^t + \frac{c}{2}\alpha_3^t$. In step (A.34c), I have used the inductive hypothesis that the bounds (A.33) hold at time t, ensuring in particular that $\alpha_3^t \leq \alpha_1^t + 1$. Finally, in step (A.34d), I have used the fact that α_1^t is an integer and $\frac{c}{2+c}$ is smaller than 1.

(A.34) ERC 1 has triggered an update between times t and $t+1 \Longrightarrow$

$$\begin{aligned}
&\overset{(a)}{\Longrightarrow} \theta_1^t \leq \theta_2^t \\
&\overset{(b)}{\Longrightarrow} c\alpha_1^t + \frac{c}{2}\alpha_2^t \leq -\alpha_1^t + \frac{c}{2}\alpha_2^t + \frac{c}{2}\alpha_3^t \\
&\Longrightarrow \alpha_1^t + c\alpha_1^t \leq \frac{c}{2}\alpha_3^t \\
&\overset{(c)}{\Longrightarrow} \alpha_1^t + c\alpha_1^t \leq \frac{c}{2}(\alpha_1^t + 1) \\
&\Longrightarrow \frac{2+c}{2}\alpha_1^t \leq \frac{c}{2} \\
&\Longrightarrow \alpha_1^t \leq \frac{c}{2+c} \\
&\overset{(d)}{\Longrightarrow} \alpha_1^t = 0
\end{aligned}$$

The chain of implications in (A.34) says that, if ERC 1 has triggered an update in between times t and $t+1$, then it has triggered no updates up until time t, namely $\alpha_1^t = 0$. As $\alpha_1^{t+1} = \alpha_1^t + 1 = 0 + 1 = 1$, the bound (A.33a) holds at time $t+1$.

Next, let me prove that the bound (A.33b) holds at time $t+1$. Again, the bound trivially holds if it is not ERC 2 which has triggered the update in between times t and $t+1$. Otherwise, I can reason as in the chain of implications in (A.35), analogous to the one in (A.34). The crucial step (*) holds because of the assumption that c is strictly smaller than 1, so that the inequality $\alpha_2^t \leq c$ (together with the fact that α_2^t is an integer) entails that $\alpha_2^t = 0$.

(A.35) ERC 2 has triggered an update between times t and $t+1 \Longrightarrow$

$$\begin{aligned}
&\Longrightarrow \theta_1^t \leq \theta_3^t \\
&\Longrightarrow c\alpha_1^t + \frac{c}{2}\alpha_2^t \leq -\alpha_2^t + \frac{c}{2}\alpha_3^t + \frac{c}{2}\alpha_4^t \\
&\Longrightarrow \frac{c}{2}\alpha_1^t + \frac{c}{2}\alpha_2^t \leq -\alpha_2^t + \frac{c}{2}\alpha_3^t + \frac{c}{2}\alpha_4^t \\
&\Longrightarrow \alpha_2^t \leq \frac{c}{2}\Big\{(\alpha_3^t - \alpha_1^t) + (\alpha_4^t - \alpha_2^t)\Big\} \\
&\Longrightarrow \alpha_2^t \leq \frac{c}{2}\Big\{1 + 1\Big\} \\
&\Longrightarrow \alpha_2^t \leq c \\
&\overset{(*)}{\Longrightarrow} \alpha_2 = 0
\end{aligned}$$

The chain of implications in (A.35) says that, if ERC 2 has triggered an update in between times t and $t+1$, then it has triggered no updates up until time t, namely $\alpha_2^t = 0$. As $\alpha_2^{t+1} = \alpha_2^t + 1 = 0 + 1 = 1$, the bound (A.33b) holds at time $t+1$.

Finally, let me prove that the bound (A.33c) holds at time $t+1$. Again, the bound trivially holds for a certain k-th ERC if it is not that k-th ERC which has triggered the update in between times t

and $t+1$. Otherwise, I can reason as in the chain of implications in (A.36), analogous to the chains in (A.34) and (A.35).

(A.36) the k-th ERC has triggered an update between times t and $t+1$

$$\begin{aligned}
&\Longrightarrow \theta^t_{k-1} \leq \theta^t_{k+1} \\
&\Longrightarrow \underbrace{-\alpha^t_{k-2} + \frac{c}{2}\alpha^t_{k-1} + \frac{c}{2}\alpha^t_k}_{\theta^t_{k-1}} \leq \underbrace{-\alpha^t_k + \frac{c}{2}\alpha^t_{k+1} + \frac{c}{2}\alpha^t_{k+2}}_{\theta^t_{k+1}} \\
&\Longrightarrow \alpha^t_k - \alpha^t_{k-2} \leq \frac{c}{2}\Big\{(\alpha^t_{k+1} - \alpha^t_{k-1}) + (\alpha^t_{k+2} - \alpha^t_k)\Big\} \\
&\Longrightarrow \alpha^t_k - \alpha^t_{k-2} \leq \frac{c}{2}\Big\{1+1\Big\} \\
&\Longrightarrow \alpha^t_k - \alpha^t_{k-2} \leq c \\
&\Longrightarrow \alpha^t_k = \alpha^t_{k-2}
\end{aligned}$$

The chain of implications in (A.36) says that, if the k-th ERC has triggered an update in between times t and $t+1$, then it has triggered as many updates up until time t as the $(k-2)$th ERC, namely $\alpha^t_k = \alpha^t_{k-2}$. As $\alpha^{t+1}_k = \alpha^t_k + 1 = \alpha^t_{k-2} + 1$, then bound (A.33c) holds at time $t+1$.

Suppose for concreteness that the total number n of constraints is odd, namely $n = 2k+1$. The total number of updates made by the learner up to a certain time t is the sum $\alpha^t_1 + \alpha^t_2 + \ldots + \alpha^t_{n-1}$ of the number α^t_1 of updates triggered by the first ERC of the anti-Pater matrix (A.29), plus the number α^t_2 of updates triggered by the second ERC, and so on, as stated in (A.37a). In step (A.37b), I have used the inequalities (A.33), which ensure in particular that $\alpha^t_{2i} \leq i$ and that $\alpha^t_{2i+1} \leq i+1$

(A.37)
$$\begin{aligned}
\text{Number of updates} &\overset{(a)}{=} \big(\alpha^t_1 + \alpha^t_3 + \alpha^t_5 + \ldots + \alpha^t_{n-2}\big) + \big(\alpha^t_2 + \alpha^t_4 + \alpha^t_6 + \ldots + \alpha^t_{n-1}\big) \\
&= \sum_{i=0}^{k-1} \alpha^t_{2i+1} + \sum_{i=1}^{k} \alpha^t_{2i} \\
&\overset{(b)}{\leq} \sum_{i=0}^{k-1} (i+1) + \sum_{i=1}^{k} i \\
&= 1 + k + (k-1) + 2\sum_{i=1}^{k-1} i \\
&= k(k+1) \\
&= \frac{1}{4}(n-1)(n+1)
\end{aligned}$$

The case where n is even is treated analogously. ■