An economy-based amendment to learning hidden structure with Robust Intepretive Parsing

Eunsun Jou

May 20, 2022

1 Introduction: Learning in the face of structural ambiguity

While many of the observable aspects of the world's languages, especially spoken ones, are linear, many current theories of linguistics assume a non-linear structural representation of language. This is true at various levels of representation of language, including syntactic constituency, lexical representations, and metrical phonology. A linear linguistic expression can correspond to multiple structural representations, leading to structural ambiguity. Structural ambiguity poses an interesting challenge to the language learner. How can a learner acquire the correct structural representation when she only has access to a linear string of linguistic symbols?

One of the ways researchers have used to investigate this topic is to computationally model the process of learning in the face of such structural ambiguity. Unlike human learners, it is much easier to probe into the process of a computational algorithm learning a language. Computational modelling also facilitates fine-tuning the algorithm's learning process, thereby enabling a controlled investigation into how the learning result is affected by various factors of the learning process.

A pertinent line of research in this area makes use of a learning process called Robust Interpretive Parsing (RIP), often implemented through computational learning algorithms such as the Gradual Learning Algorithm (GLA; Boersma 1997). RIP enables the learner to acquire the hidden structure behind observable linguistic expressions (see section 2.2 for a detailed explanation). Upon observing a linguistic expression (*e.g.,* a stress pattern of a word), the learner reverse-engineers the input of said expression and calculates what the optimal structural representation (*e.g.,* the foot structure) of that input is, based on its current (limited) knowledge about the grammar of the language. If the calculated structure is not compatible with the actual observed expression, it registers that it has run into an error. Instead of giving up, the learner searches for an alternative structural representation that is both compatible with the observed expression and as compatible as it can be with its current knowledge of the grammar. Upon choosing an alternative structural representation (called the *target parse* in this paper), the learner adjusts its grammar so that the target parse becomes the actual optimal output of the grammar.

A merit of RIP is that it is generalizeable across various linguistic phenomena, different representations of the grammar, and learning methodologies (see Jarosz 2013 for an extensive overview). While a large number of related studies apply RIP to learning metrical stress (Tesar and Smolensky 2000, Apoussidou and Boersma 2003, Boersma and Pater 2008, Jarosz 2013), it has been also applied to learning hidden lexical representations (Apoussidou 2006). In learning metrical phonology, RIP has been used within the context of Optimality Theoretic (OT) as well as Harmonic Grammar (HG). It is compatible with gradual constraint-ranking learning algorithms such as the GLA as well as non-gradual ones like Error-Driven Constraint Demotion (EDCD; Tesar and Smolensky 1998, 2000).

While the RIP is versatile, not all RIP learning is up to par in terms of performance. This is especially true with OT-based algorithms such as the RIP/OT-GLA. For the 124 constructed languages tested by Boersma and Pater (2008), the RIP/OT-GLA showed a success rate of 58.95% across 10 runs. The current study reports a higher performance rate at 77.37% for 68 randomly sampled languages (see section 6.1), but this result is still far from perfect. Despite the unsatisfactory performance, there has not been much investigation into what a failed learning trial looks like, and why it occurs. A majority of the responses to the low performance of the RIP/OT-GLA was to explore ways to avoid the failure with alternative representations of the grammar (Boersma and Pater's (2008) RIP/HG-GLA) or a stochastic resampling of the constraint rankings (Jarosz 2013). However, there are rarely any reports or a detailed analysis of what the failure itself looks like.[1] In order to gain insight about the nature of the learning performed by RIP, it is important to accurately understand the profile of failed learning trials.

It is this gap in the literature that the current paper aims to fill.[2] This paper discusses in detail examples of failed learning trials of the RIP/OT-GLA, and highlights a concrete algorithmic problem. The RIP/OT-GLA is an error-driven learning algorithm which attempts to correct its constraint ranking when it registers that it has made an error. When it corrects the grammar, it needs to choose a direction towards which it makes the correction – which constraints will it promote, and which ones will it demote? Under the traditional configuration of the RIP/OT-GLA, its choice of direction is very much dependent on the current grammar's constraint ranking. But because this ranking is necessarily flawed in some way (otherwise it would not have incurred an error), RIP/OT-GLA sometimes fails to target the crucial constraints. I report concrete examples of such problems arising in learning trials, and suggest an alternative method of choosing a direction of change that works around the problem yet still maintaints the discrete constraint-ranking spirit of the OT-GLA.

This alternative method, which I call the ERC method, opts for the most economical change. Making use of the *Elementary Ranking Condition* representation (ERC; Prince 2002), it calculates for each potential direction how many rank changes are needed for the offending constraint to be dominated. Then it chooses the direction which, if chosen, will involve the least amount of rank changes. Since the most economical

[1] A recent exception is Magri and Storme (2020), who make a detailed report on failed attempts to learn variation in Ilokano metathesis using various constraint ranking algorithms.

[2] The code used for this study, as well as the results of running the code, are available in the following repository: https://github.com/EunsunJou/Economy_RIPGLA

change does not necessarily satisfy the top-ranked constraint, it avoids the pitfall that the original RIP/OT-GLA can fall into.

The remainder of the paper is organized as follows. Section 2 provides background about the GLA and RIP. In section 3 I explain the metrical stress system, first defined by Tesar and Smolensky (2000), which I use to create the metrical stress languages against which I test my algorithm. Section 4 introduces the main puzzle addressed in this study. I explain how the way the RIP/OT-GLA currently chooses its target parse is problematic. After a brief conceptual explanation of the puzzle, I demonstrate in sections 4.1 and 4.2 how the problem actually arises in concrete learning examples. As a solution to this problem, I propose an alternative way of choosing the target parse in section 5. In section 6 I discuss the result of implementing the alternative method. While it did not significantly improve the overall success rate of the RIP/GLA, it does result in better learning for the specific cases described in sections 4.1 and 4.2. Lastly, I point out in section 6.2 that the alternative method leads to more efficient convergence in successful learning trials. Section 7 concludes.

2 The Gradual Learning Algorithm and Robust Interpretive Parsing

2.1 The Gradual Learning Algorithm

In this section I provide some background about the learning algorithm used in this study. The Gradual Learning Algorithm (GLA) is a stochastic model of a learner trying to acquire an OT grammar (Boersma 1997, Boersma and Hayes 2001). It is presented with a set of legal words and a set of constraints, and its goal is to produce a ranking of the constraints that is compatible with all the words of the language. More specifically, it is provided with the following four elements.

(1) a. *The list of words to be learned*
This is the set of words that the learner aims to be able to produce at the end of its learning process. If it comes up with a constraint ranking that is compatible with all of the words in this list, it is deemed successful.

b. *A set of constraints, and their initial ranking values* (*RVs*)
The ultimate goal of the GLA is to rank the constraints so that the ranking is compatible with all of the target words. Each constraint is linked to a numerical value, or its *ranking value* (RV). The larger the RV of a constraint is, the higher the constraint is ranked. The GLA adjusts the constraint ranking by updating each contraint's RV. In many studies, the constraints all start out with the same initial RV.

c. *A set of input-output pair* Every word in the language is the most optimal output of some input. Therefore, there are as many inputs as there are words in the language. The GLA is given information about which outputs are the potential realizations of which input.

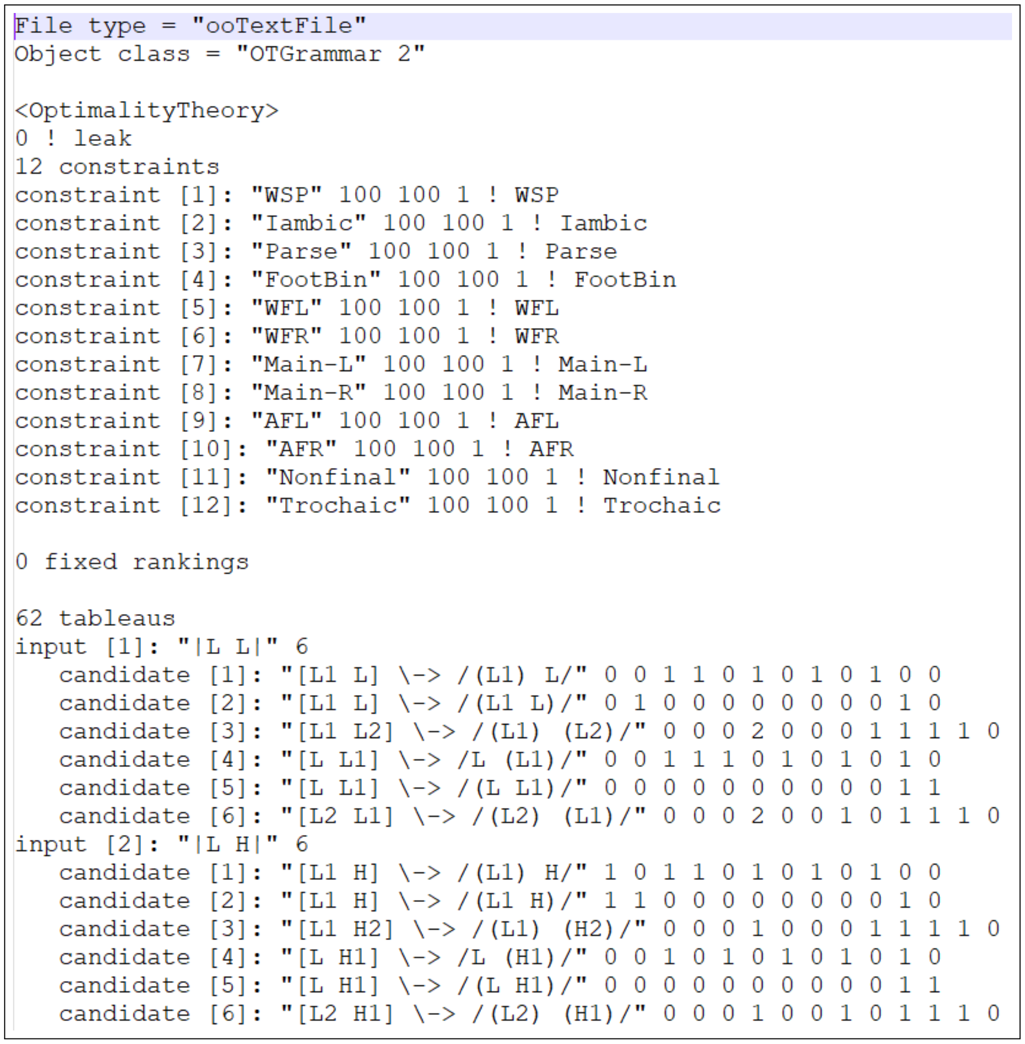
d. *An unranked tableau for the input of each target word*
The GLA is provided with unranked tableaux for each input of the language. The candidates of

each tableau are the possible realizations (outputs) of the input. The tableau also provides information about the violation profile of each candidate (*i.e.,* how many times a candidate violates each constraint). Below is a schematized example of an unranked tableau.

/Input/	Const 1	Const 2	Const 3	Const 4
a. [Output A]	0	1	0	2
b. [Output B]	1	1	3	2
c. [Output C]	0	0	2	2
d. [Output D]	0	0	0	4

Using the software Praat, the researcher can provide the GLA the information in (1b–d) in a single text file (Boersma and Weenink 2022). Figure 1 shows a partial image of such a text file. The initial RV of all constraints is 100.

Figure 1: A templatic metrical stress grammar produced with Praat



```
File type = "ooTextFile"
Object class = "OTGrammar 2"

<OptimalityTheory>
0 ! leak
12 constraints
constraint [1]: "WSP" 100 100 1 ! WSP
constraint [2]: "Iambic" 100 100 1 ! Iambic
constraint [3]: "Parse" 100 100 1 ! Parse
constraint [4]: "FootBin" 100 100 1 ! FootBin
constraint [5]: "WFL" 100 100 1 ! WFL
constraint [6]: "WFR" 100 100 1 ! WFR
constraint [7]: "Main-L" 100 100 1 ! Main-L
constraint [8]: "Main-R" 100 100 1 ! Main-R
constraint [9]: "AFL" 100 100 1 ! AFL
constraint [10]: "AFR" 100 100 1 ! AFR
constraint [11]: "Nonfinal" 100 100 1 ! Nonfinal
constraint [12]: "Trochaic" 100 100 1 ! Trochaic

0 fixed rankings

62 tableaus
input [1]: "|L L|" 6
   candidate [1]: "[L1 L] \-> /(L1) L/" 0 0 1 1 0 1 0 1 0 1 0 0
   candidate [2]: "[L1 L] \-> /(L1 L)/" 0 1 0 0 0 0 0 0 0 0 1 0
   candidate [3]: "[L1 L2] \-> /(L1) (L2)/" 0 0 0 2 0 0 0 1 1 1 1 0
   candidate [4]: "[L L1] \-> /L (L1)/" 0 0 1 1 1 0 1 0 1 0 1 0
   candidate [5]: "[L L1] \-> /(L L1)/" 0 0 0 0 0 0 0 0 0 0 1 1
   candidate [6]: "[L2 L1] \-> /(L2) (L1)/" 0 0 0 2 0 0 1 0 1 1 1 0
input [2]: "|L H|" 6
   candidate [1]: "[L1 H] \-> /(L1) H/" 1 0 1 1 0 1 0 1 0 1 0 0
   candidate [2]: "[L1 H] \-> /(L1 H)/" 1 1 0 0 0 0 0 0 0 0 1 0
   candidate [3]: "[L1 H2] \-> /(L1) (H2)/" 0 0 0 1 0 0 0 1 1 1 1 0
   candidate [4]: "[L H1] \-> /L (H1)/" 0 0 1 0 1 0 1 0 1 0 1 0
   candidate [5]: "[L H1] \-> /(L H1)/" 0 0 0 0 0 0 0 0 0 0 1 1
   candidate [6]: "[L2 H1] \-> /(L2) (H1)/" 0 0 0 1 0 0 1 0 1 1 1 0
```

It should be clarified that the researcher needs to make a decision on which candidates the GLA considers as competitors. In theory, the function *Gen* can produce many more output candidates than the ones that the GLA is given. However, since the GLA needs to be given the violation profile of all the candidates that it considers, it is difficult for the researcher to provide this information for a very large number of candidates. Therefore, it is reasonable to limit the set of candidates considered to those that have at least some chance of becoming the optimal winner under some ranking of the constraints. In this study, I simply followed the decisions made in the templatic grammar provided by Praat. It considers 6 possible candidates for 2-syllable words, 24 candidates for 3-syllable words, 88 candidates for 4-syllable words, and 300 for 5-syllable words.

With this information at hand, the GLA goes through the list of words and tries to identify the constraint ranking that is compatible with the words that it sees. Below is a description of its working process.

(2) *Learning process of the GLA*

a. The GLA reads in a word from the list, and finds the unranked tableau of which the word is a candidate output.

b. If noise is turned on, the GLA adds temporary noise to the RV of each constraint. If noise is turned off, nothing is done at this step. (See below on noise.)

c. The constraints are ranked based on the current RVs. The larger the RV, the higher the rank. Below is a schematized example.

Constraint	C_1	C_2	C_3	C_4
RV	98	101	93	102

$\Leftrightarrow \quad C_4 \gg C_2 \gg C_1 \gg C_3$

d. The GLA applies the ranking from step (2c) to the unranked tableau from step (2a). This yields a unique optimal output from the tableau.

e. The GLA compares the optimal output with the word that it encountered in step (2a). If the output is identical to the actual word, the GLA does not change any of the RVs. It returns to step (2a) to read in the next target word. If the output is different from the actual word, the GLA has run into an error. In this case it proceeds to step (2f).

f. The GLA compares the violation profile of the output with that of the word, and does two things. It promotes constraints that favor the actual word over the output, and demotes constraints that favor the output over the actual word. The amount of promotion/demotion is called the *plasticity* or the *learning rate*. It is a fairly small amount – usually 1 or 0.1. Below is a schematized example with plasticity 1.

Constraint	C_4	C_2	C_3	C_1
RV	102	101	98	93
Winner			*	*
Target Word		*	*	

$\Rightarrow$

Constraint	C_4	C_2	C_3	C_1
RV	102	**100**	98	**94**
Winner			*	*
Target Word		*	*	

After completing this step, the GLA returns to step (2a) to read in the next target word.

The GLA loops through through steps (2a–f) until it reaches the end of the target word list. I call the process of looping through the word list a *learning trial*. The set of RVs that emerge at the end of a trial is the resulting grammar of the GLA. After each trial comes the *evaluation* of said trial. Evaluation happens by applying the resulting grammar to the unranked tableaux of the language and seeing whether the resulting optimal outputs are identical to the actual words. If the outputs for all tableaux are actual words of the langauge, the learning process is deemed successful.

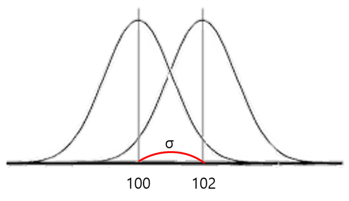
A note on noise is in order before concluding this section. Noise is a small numerical amount added to each RV before ranking the constraints and figuring out the winner from an unranked tableau. The exact amount is determined stochastically for each constraint: noise is a variable that follows a Gaussian distribution with mean 0. When a GLA is configured to "have a noise value of 2", it is understood that the noise follows a Gaussian distribution of mean 0 and standard deviation of 2. The result of adding noise to an RV is called the *selection point*. The selection point of a constraint would be a variable that follows a Gaussian distribution of which the mean is equal to the RV and the standard deviation is the noise value.

Noise is useful in modelling variation in language. Consider two constraints C_1 and C_2 whose RVs are 100 and 102 respectively. Assume a GLA with noise value of 2. While $C_2 \gg C_1$ according to their RVs, their ranking based on selection points can be easily flipped to $C_1 \gg C_2$ since their RVs are only one standard deviation apart. This is reflected in the large overlap of the two curves in figure 2a. A language that shows variation in the ranking between C_1 and C_2 would look like this.

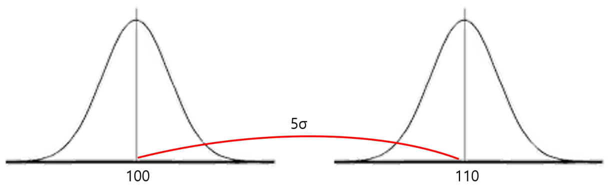
Compare this situation with one where the RV of C_1 is 100 and that of C_2 is 110. Now, with a distance of 5 standard deviations between the two means, there is a much lower chance that the ranking be flipped. Hence the small overlap in probability space in figure 2b. This distribution corresponds to a langauge that does not show variation in the ranking of C_1 and C_2. C_2 is reliably ranked above C_1, so any candidate which is favored by the ranking $C_1 \gg C_2$ would not surface in this language.

Figure 2: The distribution of selection points for constraints C_1 and C_2

(a) RV of C_1 is 100, RV of C_2 is 102



(b) RV of C_1 is 100, RV of C_2 is 110



2.2 Robust Interpretive Parsing

A modified version of the GLA has been adopted in simulating learning in the face of structural ambiguity. This is a GLA that does Robust Interpretive Parsing (RIP). In this paper I call it the RIP/OT-GLA, since I specifically focus on OT-based GLA performing RIP. The RIP/OT-GLA is *robust* in the sense that it does not crash in the face of data that is incompatible with its current grammar.

The tableau in (3) shows a hypothetical grammar that the RIP/GLA is working on. The definition of the constraints are given in the constraints are defined in (4). It aims to learn the stress pattern of Polish, and one of the words that it runs into is [tɛˈlɛfɔn]. Notice that the candidates a-d contain structural information about the foot structure of a word, while that information is missing the presented word. That is, the learner simply observes [tɛˈlɛfɔn], not /(tɛˈlɛ)fɔn/ or /tɛ(ˈlɛfɔn)/. This is characteristic of learning problems with structural ambiguity, like metrical structure. The RIP/OT-GLA is only presented with information about the observable stress pattern, but there are multiple foot structures that could have resulted in the observed stress pattern. The algorithm is faced with the task of not only coming up with a ranking that yields the correct stress pattern, but also the correct metrical structure underlying the stress pattern. The situation is parallel for human learners: humans hear stress patterns, but not foot structure. But the underlying assumption is that information about metrical structure of words is instrumental in producing the correct stress pattern.

(3) Learner's hypothetical grammar when presented with [tɛˈlɛfɔn] (Jarosz 2013:32)

|tɛlɛfɔn|	AllFt-R	Iambic	Trochaic	AllFt-L
a. /(ˈtɛlɛ)fɔn/	*	*		
b. /(tɛˈlɛ)fɔn/	*		*	
W c. /tɛ(ˈlɛfɔn)/		*		*
L d. /tɛ(lɛˈfɔn)/			*	*

(4) a. AllFeet-Right (AllFt-R) : Align each foot with the right edge of the word.
b. AllFeet-Left (AllFt-L) : Align each foot with the left edge of the word.
c. Iambic : The final syllable of a foot must be the head.
d. Trochaic: The initial syllable of a foot must be the head.

I first make a clarification about the notation of input, parse, and output in this tableau the context of RIP. The string of segments and the stress pattern that the learner observes is the output, and is written between square brackets: [tɛˈlɛfɔn]. The input corresponding the output is simply the string of segments devoid of any stress information. It is written between vertical bars: |tɛlɛfɔn|. Lastly, the parse of the word, which is the structural representation of the output, is written between slashes: /tɛ(ˈlɛfɔn)/.

Let us now study (3) in detail to better understand how the RIP/GLA performs this learning task. The current ranking that the RIP/OT-GLA has is AllFt-R ≫ Iambic ≫ Trochaic ≫ AllFt-L. Therefore, it would conclude that the optimal output of the input /tɛlɛfɔn/ is candidate d, [tɛ(lɛˈfon)]. However, the winner is clearly the wrong one because the actual target word that it observed has a different stress contour: [tɛˈlɛfɔn]. The word should have penultimate stress, not final stress. Once the RIP/OT-GLA registers that it has run into an error, it tries to figure out what the winner should have been. *However, there are two candidates that are compatible with the target word*: candidates b ([(tɛˈlɛ)fɔn]) and c ([tɛ(ˈlɛfɔn)]). The two candidates parse the target word in two different ways, but both parses result in penultimate stress. Now the GLA needs to make a choice: will it consider candidate b as its target parse, or candidate c? In making this decision, it chooses the more optimal candidate under the current ranking. In this case, it would be candidate

c. The violation of the highest-ranked constraint AllFeet-R by candidate b is critical in competition to candidate c. Once it has chosen the target parse, the RIP/OT-GLA promotes constraints that favor candidate c over candidate d (Trochaic), and demotes constraints that favor candidate d over candidate c (Iambic).

Notice that whether the algorithm chooses candidate b or candidate c as its target parse is crucial in determining the direction of learning. Since it chose candidate c as its target parse, Iambic is demoted and Trochaic is promoted. Had candidate b been chosen as the target parse, the RVs of Iambic and Trochaic would not have changed. Instead, the RIP/GLA would have demoted AllFt-R and promoted AllFt-L. Depending on which parse it thinks is correct, its direction of learning varies widely. The reason that the RIP/OT-GLA chooses candidate c is because it is configured to choose its target parse based on its current constraint ranking (Boersma 1997, Boersma and Hayes 2001). If the algorithm was configured to choose its target parse based on a different criterion, then the learning results could be different – possibly leading to better (or worse) learning results. Exploring an alternative criterion is the research question of this paper. I explain in section 4 the specific alternative criterion that I implemented and tested.

3 Learning metrical stress with the RIP/GLA

Any research project which investigates RIP learning needs some specific language with structural ambiguity for the computational learner to work on. This paper works with a system of abstract aritificial languages with metrical stress, first created by Tesar and Smolensky (2000). Previous studies, including Boersma and Pater (2008) and Jarosz (2013), have also worked with this system. It includes the following twelve constraints.

- FootBin: Each foot must be either bimoraic or bisyllabic.
- WSP: Weight-to-stress principle. Each heavy syllable must be stressed.
- Parse: Each syllable must be footed.
- Main-Right: Align the head-foot with the word, right edge.[3]
- Main-Left: Align the head-foot with the word, left edge.
- All-Feet-Right: Align each foot with the word, right edge.
- All-Feet-Left: Align each foot with the word, left edge.
- Word-Foot-Right: Align the word with some foot, right edge.
- Word-Foot-Left: Align the word with some foot, left edge.
- Iambic: Align each foot with its head syllable, right edge.
- Trochaic: Align each foot with its head syllable, left edge.[4]

[3]What is called Main-Left and Main-Right here are also named Align-Head-L and Align-Head-R. See Kager (2007:210) for an example of using Align-Head-R in explaining data from Cairene Arabic.

[4]Tesar and Smolensky use the constraint FootNonfinal which requires the head syllable to not only be aligned to the left edge, but to have a syllable follow after it. In other words, a foot such as (H1) would satisfy Trochaic but it would not satisfy FootNon-

• NONFINAL: Do not foot the final syllable of the word.

When the constraints are each assigned some RV, and hence form a ranking, they will be consistent with a *language*. A language in this system is a set of abstract *words*, which in turn are abstract *inputs* bearing a stress pattern. The abstract inputs are sequences of light or heavy syllables (represented as L and H, respectively). The length of an input is 2 syllables at the shortest, and 7 syllables at the longest. The 2-5 syllable inputs are all possible combinations of L and H, which result in a total of 60 kinds of inputs. In addition to these, there are the 6-syllable and 7-syllable inputs which exclusively consist of L's: |L L L L L L|, and |L L L L L L L|. Each language assigns different stress patterns to its inputs, resulting in a unique set of 62 words. See appendix A for the full list of words from two sample languages.

Since this paper aimed to closely examine individual learning trials, especially failed ones, it was necessary to create a fairly large set of languages. I followed the following process to obtain this set. The process ensures that the languages are mutually different, and are OT-consistent (consistent with at least one constraint ranking).

(5) *Process of randomly sampling metrical stress languages*

a. Starting with 100 grammar templates, I assigned a random integer between 80-120 to each of the 12 constraints as their ranking values, thereby creating 100 grammars. The 80-120 range is based on observations from pilot trials that the ranking values resulting from successful trials are typically within this range.
b. From the 100 grammars, I picked out the grammars that were non-varying (having only one optimal output per input). I judged a grammar as non-varying if for 100,000 generations per input, there was only one optimal output generated. This process was facilitated by use of a Praat script (Boersma and Weenink 2022).
c. Each set of outputs generated from a grammar now constitute a language. Since there are 62 kinds of inputs (from |L L|, |L H|, ... to |L L L L L L L|), each language has 62 different words.
d. Multiple grammars can generate the same language. So I picked out unique languages from my set of non-varying languages.

This process yielded 68 unique non-varying languages. They were simply named *Language 1*, *Language 2*, ..., *Language 68*. In the remainder of the paper I refer to the languages as *Lang 1*, *Lang 2*, ..., *Lang 68* for brevity.

With these 68 languages, I now had ample learning problems for the RIP/GLA to solve. I implemented the RIP/GLA in the language Python, and put this implementation to the task of learning the 68 languages. For each language, the RIP/GLA did 30 learning trials, each trial followed by an evaluation. As a result, I

FINAL. Their reason for this choice is to reflect a typological asymmetry pointed out Hayes (1995): trochaic languages may be either quantity sensitive or insensitive, while iambic languages are always quantity sensitive. I judged that this typological generalization is not very relevant for the artificial languages that I have been working with and thus chose the constraint TROCHAIC, which is the mirror image of IAMBIC. The violation profiles of candidates are also adjusted appropriately, since the grammar templates are automatically produced by Praat and it gives the user the choice between TROCHAIC and FOOTNONFINAL.

had access to a total of 2040 learning results, with information about the resultant constraint ranking as well as the errors produced at evaluation. After careful examination of individual learning results, I was able to find actual empirical instantiations of a conceptual puzzle raised by Jarosz (2013). I elaborate on this finding below.

4 The puzzle: Choosing the target parse with a losing grammar

I have highlighted at the end of section 2.2 that choosing the correct target parse is crucial in determining the learning trajectory of the GLA. Under the way it is configured, the RIP/OT-GLA uses its current OT grammar to choose its target parse. Among the candidates that are compatible with the observed stress contour, the one that is *most optimal* under the current OT grammar is chosen as the target parse. Jarosz (2013) points out that this way of choosing a target parse is problematic because it is equivalent to choosing with a "losing grammar". By the point the RIP/OT-GLA registers that it has run into an error, the constraint ranking that it has is guaranteed to be wrong one. Otherwise it wouldn't have run into an error. However, it continues to parse the observed stress pattern with its current, decidedly wrong grammar.

Jarosz acknowledges this problem at a conceptual level, but there have not been empirical demonstrations of this problem actually affecting the results of learning. In this section, I demonstrate actual examples of the RIP/OT-GLA running into a problem due to its OT-based mechanism of choosing the target parse.

4.1 Case study 1: Lang 55

Out of the 30 learning trials for Lang 55, the RIP/OT-GLA succeeded in 9 of them. We know that Lang 55 is a language with strictly binary, trochaic feet. This is because all of the grammars resulting from the 9 successful trials showed this property: they rank FootBinary at or near the top, and rank Trochaic over Iambic (Trochaic $\gg$ Iambic). Another property that differentiated the successful and failed grammars is the ranking between WordFoot-Right (WFR) and Main-Left (Main-L). In the grammars resulting from the 9 successful trials, WFR was ranked above Main-L (WFR $\gg$ Main-L). For the 21 failed trials, WFR was ranked below Main-L (WFR $\ll$ Main-L).

Now that we have a sense of what characterizes the successful and failed learning trials, let us examine what made the failed trials fail. There was a clear pattern throughout 20 of the 21 failed trials: the algorithm failed to learn the correct stress pattern for the input |L L L|, which is [L L1 L].[5] (The number 1 corresponds to main or primary stress. The L1 here is the syllable bearing primary stress. Sometimes words also have secondary stress, which is marked with the number 2.) In these failed trials, the RIP/GLA produced [L1 L

[5]The RIP/OT-GLA, or at least my implementation of it, occasionally "crashes". The one exceptional trial here was a crashed trial. Crashed trials are characterized by extremely low RVs and a drastic failure to learn. At the end of this trial, the RVs of some constraints plunged under -7000 and the algorithm never learned 50 out of the 62 words. These learning trials are qualitatively different from the normal but failed trials. I do not have an explanation for why these crashes happen, but they are rare enough that they do not get in the way of observing the typical cases of failure. Out of the 29 failed trials of Lang 3 (discussed in section 4.2), there were no crashed trials.

L], with a left-aligned trochee: /(L1 L) L/. In the face of this error, which target parse compatible with [L L1 L] would the RIP/GLA choose?

Figure 3 shows an example of a grammar from a *failed* trial, and how the three candidate RIP parses (candidates b-d) would fare against the wrongly produced parse /(L1 L) L/ (candidate a). According to the tableau, the RIP/GLA would choose candidate b as its target parse since it is the most optimal under the current ranking. Upon choosing candidate b as its target parse, the RIP/OT-GLA promotes constraints that favor candidate b over candidate a and demotes constraints that favor the latter over the former. This means that TROCHAIC will be demoted and IAMBIC will be promoted. But we know from observing the 9 successful trials that for the learning to be successful, TROCHAIC needs to outrank IAMBIC. Furthermore, we know that WFR should outrank MAIN-L for the learning to be successful. But since candidate b and candidate a agree on both constraints, the ranking between these two constraints will be left untouched.

Figure 3: A tableau from a failed learning trial of Lang 55

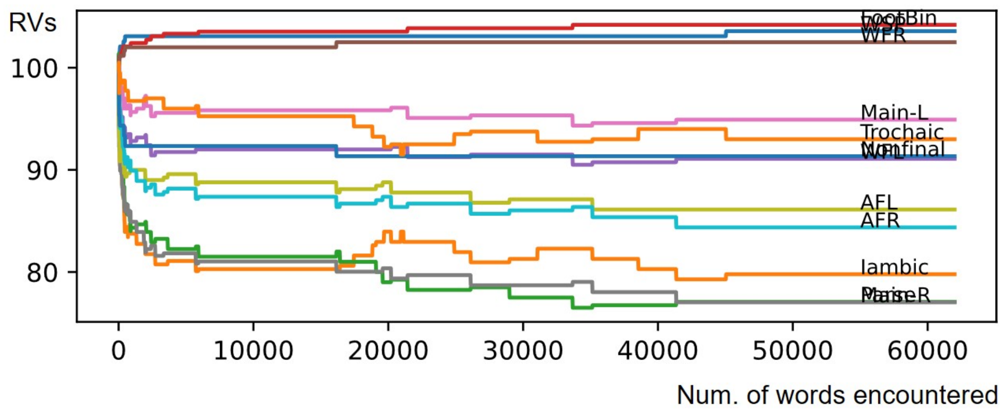
|L L L|	FTBIN	WSP	ML	WFR	WFL	MR	TROC	AFR	AFL	NONFIN	IAMB	PARSE
☞ a. /(L1 L) L/	0	0	0	1	0	1	0	1	0	0	1	1
b. /(L L1) L/	0	0	0	1	0	1	1	1	0	0	0	1
c. /L (L1 L)/	0	0	1	0	1	0	0	0	1	1	1	1
d. /L (L1) L/	1	0	1	1	1	1	0	1	1	0	0	2

As a result of choosing candidate b, the RIP/GLA focuses on the wrong pair of candidates to work on: it promotes IAMBIC and demotes TROCHAIC. Upon running into other words, it demotes IAMBIC and promotes TROCHAIC because that is what the overall stress pattern of the langauge dictates. In a later stage, when it runs into [L L1 L] again, it does the same thing – promoting IAMBIC and demoting TROCHAIC. All the while, the constraints which actually hinder the RIP/OT-GLA from successfully learning the grammar – WFR and MAIN-L – remain untouched. In a way, this can be understood as the RIP/OT-GLA falling into some sort of local optimum and not being able to escape it, thereby failing to reach the global optimum.

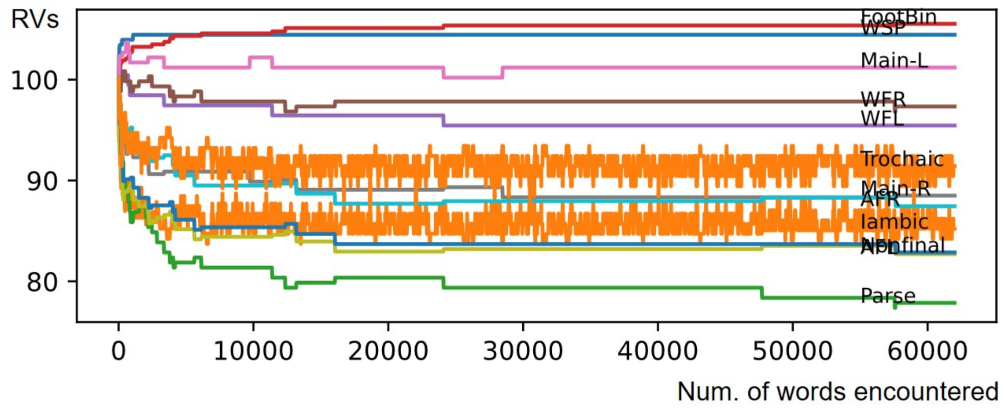
Figure 4 effectively visualizes the problem that it has fallen into. The horizontal axis represents the entire length of the trial. The vertical axis represents the range of ranking values. Each line corresponds to the trajectory of the ranking value of a constraint; the constraint names are labelled on the lines. In a successful learning trial of Lang 55 (and of all languages in general), the ranking values stabilize after about 10,000 words. However, in the problematic failing trials, the RVs of IAMBIC and TROCHAIC continuously oscillate. Note that the RVs of WFR and MAIN-L stabilize in the wrong order and are never flipped. And at the end of an unsuccessful trial, the RIP/GLA produces a grammar that predicts the wrong stress pattern for /L L L/.

Figure 4: Ranking value trajectories of a successful and failed learning trial of Lang 55

(a) A successful learning trial



(b) A failed learning trial



4.2 Case study 2: Lang 3

Lang 3 is another language that the RIP/OT-GLA had difficulty learning. It failed in 29 out of the 30 trials for this language. Similarly to how it kept failing on the same input (|L L L|) for Lang 55, it kept failing to learn the same set of words. But this time, there were two different sets of words that kept causing it to fail. In 15 of the 29 trials, it failed on the inputs |H L L L| and |H H L L L|. In the remaining 14, it failed on three inputs: |L L L L L|, |L H L L L|, and the seven-syllable |L L L L L L L|. This suggests that there are two distinct groups of wrong grammars that the RIP/OT-GLA arrived at. In this section I illustrate a problem with the former set of grammars, which produced wrong outputs for |H L L L| and |H H L L L|. Below I present information about the two inputs in Lang 3.

(6) a. Correct output of /H L L L/: [H1 L L L2]
The learner's wrong output: [H1 L L2 L] (foot structure /(H1) (L L2) L/)

b. Correct output of /H H L L L/: [H1 H2 L L L2]
The learner's wrong output: [H1 H2 L L2 L] (foot structure /(H1) (H2) (L L2) L/)

Crucially, the 15 grammars which produced these wrong outputs rank IAMBIC over PARSE (IAMBIC $\gg$ PARSE), while the correct trial resulted in the opposite ranking (PARSE $\gg$ IAMBIC).[6] Figures 5a and 5b are tableaux from a failed learning trial of Lang 3.

Figure 5: Tableaux from a failed learning trial of Lang 3

(a) Tableau for |H L L L|

H L L L	FtBin	WSP	ML	WFL	Iamb	Parse	MR	Troch	NonFin	WFR	AFL	AFR
☞ a. /(H1) (L L2) L/	0	0	0	0	0	1	3	1	0	1	1	4
b. /(H1) L (L L2)/	0	0	0	0	0	1	3	1	1	0	2	3
c. /(H1 L) (L L2)/	0	0	0	0	1	0	2	1	1	0	2	2
d. /(H1 L) L (L2)/	1	0	0	0	1	1	2	0	1	0	3	2
e. /(H1) L L (L2)/	1	0	0	0	0	2	3	0	1	0	3	3

(b) Tableau for |H H L L L|

H H L L L	FtBin	WSP	ML	WFL	Iamb	Parse	MR	Troch	NonFin	WFR	AFL	AFR
☞ a. /(H1) (H2) (L L2) L/	0	0	0	0	0	1	4	1	0	1	3	8
b. /(H1) (H2) L (L L2)/	0	0	0	0	0	1	4	1	1	0	4	7
c. /(H1) (H2 L) (L L2)/	0	0	0	0	1	0	4	1	1	0	4	6
d. /(H1) (H2) L L (L2)/	1	0	0	0	0	2	4	0	1	0	5	7
e. /(H1) (H2 L) L (L2)/	1	0	0	0	1	1	4	0	1	0	5	6

In both of the tableaux, candidate a is the output produced by the RIP/OT-GLA under the current constraint ranking. Candidates b–d are the possible RIP parses. Candidates d and e can be immediately ruled out due to their fatal violation of FOOTBINARY. Between candidates b and c, the algorithm chooses candidate b because of the violation of IAMBIC by candidate c. Once it chooses the target parse as candidate b, it promotes constraints that favor candidate b over candidate a while demoting constraints that favor candidate a over candidate b. In this case, the former are WORDFOOT-RIGHT and ALLFEET-RIGHT while the latter are NONFINAL and ALLFT-LEFT. But we know from the successful learning trial that the crucial change that needs to be made is to flip the ranking between IAMBIC and PARSE. Because the target parse and the current output behave identically with regards to IAMBIC and PARSE, the RV of these two candidates are never changed.

[6]The reader may rightly be hesitant to accept that PARSE $\gg$ IAMBIC is a correct characterization of Lang 3, since there is only one successful learning trial to draw this characterization from. However, the 17 successful trials under the amended RIP/GLA with the ERC method also all ranked PARSE over IAMBIC. See section 6.1.

Hence Lang 3 demonstrates another example the RIP/OT-GLA failing to find the correct constraint ranking due to its choice of target parse.

5 The solution: An ERC-based target-choosing algorithm

The RIP/OT-GLA chooses as its target parse most optimal of the RIP parses under the current constraint ranking. In other words, the candidate which satisfies more of the higher-ranked constraints is chosen as the target parse. However, we have seen concrete examples where this criterion of choosing the target parse caused problems.

In this section I propose an alternative method of choosing a target parse. I call the alternative method the *ERC method*, and I call the RIP/GLA algorithm that uses this method the *RIP/ERC-GLA*. Instead of choosing the target parse that satisfies the higher ranked constraints, the RIP/ERC-GLA chooses the candidate parse which, if chosen as the target, would need the least number of rereankings to accommodate for. The number of rerankings is measured using the Elemantary Ranking Condition (ERC) representation first proposed by Prince (2002). In section 5.1 I briefly explain the ERC representation, and how it can be used to measure the amount of rank changes needed to make the grammar favor one parse over the other. I then explain how I applied this logic to my implementation of the RIP/ERC-GLA in section 5.2.

5.1 The Elementary Ranking Condition

The objective of error-driven learning is to change the grammar so that the resulting grammar does not produce the wrong output (the error) but instead produces the correct output. Let us call the wrong output the *loser*, and the correct output the *winner*. The ERC representation classifies constraints into three classes: *loser-preferring*, *winner-preferring*, and *even*. A loser-preferring constraint is one which is violated more by the winner than the loser. It might be the case that only the winner violates this constraint, or in the case of gradiently violable constraints the winner violates the constraint more times than the loser does. A winner-preferring constraint, similarly, is a constraint that is violated more by the loser than the winner. An even constraint is violated the same amount by both constraints.

This classification is based on a winner-loser pair. It is often the case that more than one potential winner is considered against the loser. In this case, the ERC representation is applied to each winner-loser pairing. In figure 6 I present the ERC representation of the tableau of a failed trial of Lang 55, with the tableau itself repeated for reference. The line labels in the ERC representation always correspond to the label of the candidate RIP parse in the tableau. For example, line b in figure 6 is the ERC representation of the winner-loser pair where the winner is candidate b. In the same figure, line c is the ERC representatio of the winner-loser pair where the winner is candidate c. (The loser is always candidate a.)

Figure 3: A tableau from failed learning trial of Lang 55 (repeated from section 4.1)

|L L L|	FtBin	WSP	ML	WFR	WFL	MR	Troc	AFR	AFL	Nonfin	Iamb	Parse
☞ a. /(L1 L) L/	0	0	0	1	0	1	0	1	0	0	1	1
b. /(L L1) L/	0	0	0	1	0	1	1	1	0	0	0	1
c. /L (L1 L)/	0	0	1	0	1	0	0	0	1	1	1	1
d. /L (L1) L/	1	0	1	1	1	1	0	1	1	0	0	2

Figure 6: An ERC representation of the tableau of Lang 55

	FtBin	WSP	ML	WFR	WFL	MR	Troc	AFR	AFL	Nonfin	Iamb	Parse
b. /(L1 L) L/ ∼ **/(L L1) L/**	e	e	e	e	e	e	L	e	e	e	W	e
c. /(L1 L) L/ ∼ **/L (L1 L)/**	e	e	L	W	L	W	e	W	L	L	e	e
d. /(L1 L) L/ ∼ **/L (L1) L/**	L	e	L	e	L	e	e	e	L	e	W	L

A benefit of using the ERC representation is that it makes it clear which constraints should be changed in ranking in order to accommodate the potential winner. For each winner-loser pair, it is crucial that the highest-ranked or *undominated* L (which is always the leftmost L in an ERC representation) be outranked by a W. Consider first candidate b, or /(L L1) L/. In order to change the current ranking to one that would produce candidate b as the optimal winner, Iambic needs to outrank Trochaic. Once Iambic outranks Trochaic, /(L L1) L/ would be the parse that is the most harmonic under the new constraint ranking. Similarly, for candidate c, it is important that Main-Left be outranked by some winner-preferring constraint such as Word-Foot-Right or Main-Right.

5.2 Implementing the ERC method

While any W can be promoted to outrank an L in an ERC notation, the most economical change would be to promote the highest W. For candidate c in figure 6) this would mean promoting WordFoot-Right. For the other two candidates, there is only one W, so that promoting that W is automatically the most economical change. The ERC method I propose here compares the *LW distance* of all the winner-loser pairs of an ERC representation and chooses the pair with the shortest LW distance. I define LW distance as the number of constraints the highest W would have to overcome for it to outrank the undominated L. For candidate b, the LW distance is 4 since Iambic needs to switch rankings with four constraints above it (NonFinal, AllFeet-Right, AllFeet-Left, Trochaic) in order to outrank Trochaic. For candidate c, the LW distance is 1 since WordFoot-Right only needs to overcome Main-Left. Lastly, the LW distance for candidate d is 10.

Using the ERC method, candidate c is chosen as the target parse. As a result, the RIP/ERC-GLA promotes constraints that prefer candidate c over candidate a (the current winner /(L1 L) L/), and demotes constraints that prefer candidate a over candidate c. Specifically, WordFoot-Right, Main-Right, AllFeet-Right are promoted and Main-Left, AllFeet-Left are demoted. What is important is that WordFoot-Right is promoted and MAIN-LEFT is demoted. Since this WORDFOOT-RIGHT is ranked above MAIN-LEFT in all 9 of the successful learning trials of Lang 55, this is indeed a step in the right direction. Compare this result with that of the OT-based choice of target parse. Using the OT method, the RIP/OT-GLA would choose candidate b as its target parse. Once this candidate is chosen as the target, only IAMBIC will be promoted and TROCHAIC will be demoted. Since the problematic ranking of MAIN-LEFT $\gg$ WORDFOOT-RIGHT is never fixed, the RIP/OT-GLA fails to change the grammar in a desirable direction.

Let us also examine how applying the ERC method to Lang 3 affects the direction of change. The tableaux in Figure 5 are repeated from section 4.2, followed by their ERC representation.

Figure 5: A tableau from a failed learning trial of Lang 3 (repeated from section 4.2)

(a) Tableau for |H L L L|

|H L L L|	FTBIN	WSP	ML	WFL	IAMB	PARSE	MR	TROCH	NONFIN	WFR	AFL	AFR
☞ a. /(H1) (L L2) L/	0	0	0	0	0	1	3	1	0	1	1	4
b. /(H1) L (L L2)/	0	0	0	0	0	1	3	1	1	0	2	3
c. /(H1 L) (L L2)/	0	0	0	0	1	0	2	1	1	0	2	2
d. /(H1 L) L (L2)/	1	0	0	0	1	1	2	0	1	0	3	2
e. /(H1) L L (L2)/	1	0	0	0	0	2	3	0	1	0	3	3

(b) Tableau for |H H L L L|

|H H L L L|	FTBIN	WSP	ML	WFL	IAMB	PARSE	MR	TROCH	NONFIN	WFR	AFL	AFR
☞ a. /(H1) (H2) (L L2) L/	0	0	0	0	0	1	4	1	0	1	3	8
b. /(H1) (H2) L (L L2)/	0	0	0	0	0	1	4	1	1	0	4	7
c. /(H1) (H2 L) (L L2)/	0	0	0	0	1	0	4	1	1	0	4	6
d. /(H1) (H2) L L (L2)/	1	0	0	0	0	2	4	0	1	0	5	7
e. /(H1) (H2 L) L (L2)/	1	0	0	0	1	1	4	0	1	0	5	6

Figure 7: ERC representation of RIP parse candidates of Lang 3

(a) ERC representation for |H L L L|

	FtBin	WSP	ML	WFL	Iamb	Parse	MR	Troc	NonFin	WFR	AFL	AFR
b. ∼ /(H1) L (L L2)/	e	e	e	e	e	e	e	e	L	W	L	W
c. ∼ /(H1 L) (L L2)/	e	e	e	e	L	W	W	e	L	W	L	W
d. ∼ /(H1 L) L (L2)/	L	e	e	e	L	e	W	W	L	W	L	W
e. ∼ /(H1) L L (L2)/	L	e	e	e	e	L	e	W	L	W	L	W

(b) ERC representation for |H H L L L|

	FtBin	WSP	ML	WFL	Iamb	Parse	MR	Troc	NonFin	WFR	AFL	AFR
b. ∼ /(H1) (H2) L (L L2)/	e	e	e	e	e	e	e	e	L	W	L	W
c. ∼ /(H1) (H2 L) (L L2)/	e	e	e	e	L	W	e	e	L	W	L	W
d. ∼ /(H1) (H2) L L (L2)/	L	e	e	e	e	L	e	W	L	W	L	W
e. ∼ /(H1) (H2 L) L (L2)/	L	e	e	e	L	e	e	W	L	W	L	W

In figure 7a, the LW distance of candidate b is 1, since WFR needs to outrank NONFINALITY right above it. The LW distance of candidate c is also 1, although this time it is because PARSE needs to outrank IAMB. For candidate d, the LW distance is 6: MAIN-RIGHT needs to overcome six constraints above it to be able to dominate the L of FOOTBINARY. Lastly, the LW distance of candidate e is 7. In figure 7b, the LW distance of candidates b and c are again 1. The LW distance for candidates d and e are 7 since TROCHAIC needs to overcome seven constraints. Under the ERC method, both tableaux have two winners with the shortest LW distance: candidates b and c. In this case, the RIP/ERC-GLA simply chooses one of the two randomly. If candidate b is chosen, WORDFOOT-RIGHT and ALLFEET-RIGHT are promoted while NONFINAL and ALLFEET-LEFT are demoted. If candidate c is chosen, the algorithm additionally demotes IAMBIC and promotes PARSE (and MAIN-RIGHT for |H L L L|).

It was pointed out in section 4.2 that all successful constraint rankings compatible with all of the words of Lang 3 rank PARSE over IAMBIC. Notice that the RIP/ERC-GLA is able to make the correct change by choosing candidate c as its target parse. Compare this situation with that of the RIP/OT-GLA. Under OT, the target parse will always be candidate b. As a result, the RIP/OT-GLA never has a chance to register the ranking IAMBIC $\gg$ PARSE as problematic, and fails to ever change it. Hence the ERC method makes it possible for the RIP/OT-GLA to choose a target parse that actually pushes it in the right direction. In the following section I discuss how this alternative algorithm affected the learning results of the 68 artificial languages I have sampled in this study.

6 Results and discussion

6.1 Similar performance across two methods, but improvement in Languages 55 and 3

In section 3, I explained how I implemented the RIP/OT-GLA in Python. In order to ensure that the implementation was faithful to that of the authors, I attempted to replicate some of the results of GLA learning from previous studies. I was able to replicate the results of a study by Boersma and Hayes (2001) where the GLA learns Ilokano metathesis. I was also able to replicate the learning problem illustrated by Pater (2008), whereby the GLA endlessly promotes the RV of the constraints of a specific artificial language. (See below on Magri's (2012) solution to this problem.) Having ensured that my implementation faithfully replicates previous studies, I proceeded to implement the ERC method as an addition to the code.

Once the implementation was complete, I compared the result of learning the 68 artificial languages that I sampled using the RIP/OT-GLA and the RIP/ERC-GLA. Other than the method of choosing the target parse, all other parameters were set identically. Plasticity was set at 1, and noise at 2. 30 trials were carried out for each of the 68 languages. In each trial, the algorithm encountered each of the 62 words 1,000 times, amounting to a total of 62,000 individual words seen in each trial. The order of the 62,000 words was randomized for each trial. For both methods, I adopted the calibrated promotion of constraints proposed by Magri (2012). Calibrated promotion is the solution which Magri suggested as a solution to Pater's (2008) learning problem. A learner that uses calibrated promotion demotes RVs by 1, but promotes them by the value of plasticity divided by the number of constrains to be promoted. For example, if there are two loser-preferring constraints and two winner-preferring constraints, the RV of the loser-preferring constraints are each demoted by 1 while the RV of the winner-preferring constraints are each promoted by 0.5.

At the end of the trial, there was an evaluation phase where it produced an output under its final constraint ranking for each input. Finally, the learner produced a text report of each trial where it printed out the final constraint ranking, whether it succeeded in learning the language, and which words it failed to learn in case it failed. See appendix B for an example of the printed results.

Table 1 summarizes the result of learning under the two methods.[7] The overall performances of the RIP/OT-GLA and the RIP/ERC-GLA are similar. The RIP/OT-GLA using succeeded in learning an average of 23.21 out of 30 trials per language. With the alternative ERC method, the RIP/ERC-GLA succeeded in an average of 23.80 trials.

Table 1: Mean number of successful trials (out of 30) across all languages

	OT method	ERC method
Mean	23.21 (77.37%)	23.80 (79.33%)
Standard Deviation	9.57	6.42

While the ERC method did not improve overall performance, it clearly improved the learning rate for the test case languages, Lang 55 and Lang 3. The RIP/OT-GLA made 9 successful trials of learning Lang 55 and only one successful trial learning Lang 3. In contrast, the RIP/ERC-GLA made 26 successful trials for Lang 55 and 17 successful trials for Lang 3.

Table 2: Number of successful trials (out of 30) for Lang 55 and Lang 3

	OT method	ERC method
Lang 55	9	26
Lang 3	1	17

These results suggest that there are multiple factors hindering the success of RIP/OT-GLA. One of these factors is the choice of the target parse. While adopting the ERC method does not bring about a drastic improvement for all languages, the boost in performance for Lang 55 and Lang 3 indicates that the choice of target parse is one of the prominent problems of the RIP/OT-GLA that can be addressed by adopting this method.

[7]Under both the OT and ERC methods, the overall performance is much higher than the 58.95% reported by Boersma and Pater (2008). I attribute this difference to the difference in the set of target languages learned. The large standard deviation indicates that the difficulty of a language varies widely for the RIP/GLA – some languages are learned more easily, while others are much harder to learn. The learning results are affected by the kind of language that the learner is presented with. I was not able to obtain information about the 124 constructed languages used by Boersma and Pater, but it is quite plausible that their set of languages and my set of 68 randomly sampled languages do not have much overlap between them.

6.2 More efficient convergence with the ERC method

The desidratum of the ERC method is to produce the correct stress pattern by undergoing the most economical change. It chooses as its target the parse which involves the least amount of change for the critical constraint (the highest W in ERC representation). It is then perhaps no surprise that among the successful trials, ones that involved the ERC method converged on its target grammar more efficiently than the ones using the OT method. As shown in the sample result files in appendix B, the RIP/GLA records how many times a grammar change occurred during a trial. A grammar change occurs anytime the algorithm registers an error, and performs promotion and demotion of RVs. Since it encounters 62,000 words in each trial, a learner can in theory change the grammar up to 62,000 times if it incurs an error for every word it encounters. Of course, the actual number of grammar changes are much lower than that; but the numbers still range from 14 to 30,233.

As an approximate measure of efficiency, I collected the median of the number of grammar changes of all successful trials for each language under the OT method and the ERC method. For example, there were 27 successful learning trials of Lang 6 under the OT method. The number of grammar changes ranged from 113 to 1465 for the 27 trials, and the median was 241. Under the ERC method, there were 23 successful learning trials, with grammar changes ranging from 115 to 339. The median was 153. Since the median under the ERC method was lower than that under the OT method, I deemed the ERC method as producing more efficient convergence for Lang 6.

There are two reasons why I opted for a comparison of medians. First, there are outliers which make the mean a less representative value. Due to the stochastic nature of the GLA, it sometimes takes the RIP/GLA a significantly larger amount of grammar changes than typical to achieve convergence. In learning Lang 62 under the OT method, for example, the RIP/OT-GLA typically converged after 200–300 grammar changes. But in one trial, it needed 3363 changes before converging to a grammar. In this case, the mean of the grammar changes for the 25 trials amounts to 392.64 while the median is 223, a much truer representation of the actual number of grammar changes. Second, the sample size is uneven between the two methods. As we saw for Lang 6, there were 23 successful trials under the ERC method and 27 successful trials under the OT method. Since I only collected data on convergent (successful) trials, the number of trials counted were almost always different for the two methods across all languages.

Table 3 shows the results of the comparison. There are 65 languages accounted for here. Languages 13, 34, and 35 are excluded because there were no successful learning trials under one or both of the methods, thus making a comparison impossible. Among the 65 languages compared, 45 of them showed a lower median under the ERC method. The mean difference of the median grammar changes for these languages is 946.77. In other words, it took the RIP/OT-GLA on average 946.77 more grammar changes than with the RIP/ERC-GLA to converge on the correct grammar. It is clear that for these languages, the ERC method led to more efficient convergence in comparison to the OT method.

The situation is quite different for the 20 languages which showed a smaller median grammar change with the OT method. It may seem like the OT method leads to more efficient convergence in these languages.

Table 3: Comparison of median number of grammar changes under the OT and ERC methods

(a) Languages that showed lower median change under the ERC method ($n = 45$)

Language	OT	ERC	Difference
Lang 01	21	19	2
Lang 03	1279	145	1134
Lang 05	160	158	2
Lang 06	241	153	88
Lang 07	189.5	187.5	2
Lang 09	141	133.5	7.5
Lang 10	105	97	8
Lang 11	131	89	42
Lang 12	57	54	3
Lang 14	20	20	0
Lang 15	3500	129	3371
Lang 17	53	51	2
Lang 18	146.5	90.5	56
Lang 19	19	19	0
Lang 20	170	107	63
Lang 21	5478	159	5319
Lang 22	5348.5	39	5309.5
Lang 24	93	83	10
Lang 25	16.5	16	0.5
Lang 27	954	189.5	764.5
Lang 29	3061	106.5	2954.5
Lang 32	132	129	3
Lang 33	5426	100	5326
Lang 37	1729	148	1581
Lang 38	2915	157	2758
Lang 41	451.5	43	408.5
Lang 42	50	49.5	0.5
Lang 44	9723	44	9679
Lang 45	97.5	97	0.5
Lang 46	90	86	4
Lang 48	248	229.5	18.5
Lang 49	2963	103	2860
Lang 50	96	91	5
Lang 51	51.5	49	2.5
Lang 52	274	104.5	169.5
Lang 54	280	88	192
Lang 55	231	105	126
Lang 57	184	168	16
Lang 58	232	221.5	10.5
Lang 59	19	19	0
Lang 62	223	214	9
Lang 63	141	129	12
Lang 64	117	100	17
Lang 67	119	118.5	0.5
Lang 68	414	147	267
Average difference			**946.77**

(b) Languages that showed lower median under the OT method ($n = 20$)

Language	OT	ERC	Difference
Lang 02	82.5	86	3.5
Lang 04	147	159	12
Lang 08	82.5	84	1.5
Lang 16	187	198.5	11.5
Lang 23	130	180	50
Lang 26	50	52	2
Lang 28	177	184	7
Lang 30	115	122	7
Lang 31	133.5	146	12.5
Lang 36	63	67.5	4.5
Lang 39	62	68	6
Lang 40	62.5	63	0.5
Lang 43	145	155	10
Lang 47	89	102	13
Lang 53	142.5	151.5	9
Lang 56	75	79	4
Lang 60	109	121	12
Lang 61	49	54.5	5.5
Lang 65	100	104	4
Lang 66	140.5	190	49.5
Average difference			**11.25**

However, the mean difference of the median grammar changes for these languages is 11.25. In other words, the it took the RIP/ERC-GLA on average 11.25 more grammar changes to convergence than the RIP/OT-GLA. This seems a rather insignificant difference against a backdrop of 62,000 potential opportunities for grammar change, and against the much larger difference of 946.77 in the other group. Given that the number of grammar changes can differ on the order of 100 even within the same language under the same configuration, this difference seems more like a tie rather than a significantly more efficient convergence. Overall, the comparison suggests that adopting the ERC method often leads to more efficient convergence.

The ERC method by itself does not enhance the success rate of the RIP/GLA, but it does help the learner reach its goal more quickly when the learner is generally on the right track. Since the ERC method is designed to opt for efficiency, it is not surprising that it leads to efficient convergence. However, the magnitude of the difference is telling. When the ERC method leads to more efficient convergence, it does so by an average of 946 grammar changes. The median grammar change of Languages 21, 22, and 33 are above 5,000 under the OT method but 159, 39, and 100 respectively under the ERC method. This drastic improvement suggests that there is a factor that contributes to inefficient learning, at least in certain circumstances under the OT method. I leave further investigation of this issue to future research.

7 Conclusion

This paper demonstrated a problem that arises under the original configuration of the RIP/OT-GLA. When the learner registers that it has made an error, it needs to choose an alternative parse of the word that it is currently faced with. The choice of the target parse determines which constraints are promoted and demoted, hence affecting the direction of learning. In choosing the target parse, the learner chooses the one which satisfies the current constraint ranking the most. In other words, the parse which is favored by the high-ranked constraints is chosen as the target. But this is problematic because the constraint ranking is necessarily flawed – if it was not flawed, it would not have incurred an error. This problem was previously recognized by Jarosz (2013), but at a conceptual level. This paper demonstrates that the problem is real. It discusses the case of Language 55 and Language 3, where the OT-based choice of target parse hinders the learner from changing the constraint rankings that crucially need to be changed in order to reach success. As a solution to these problems, I suggest the ERC method as an alternative way of choosing the target parse. The ERC method rewards economic change. Among the potential target parses, the learner chooses the parse which needs the least amount of rerankings in order to be produced. Since the most economic change does not always mean satisfying the high-ranked constraints, this method is less dependent on the current, flawed constraint ranking. While the ERC method is not a cure-all for all the problems of RIP/GLA, it certainly solves the problem that it is designed to solve. The learner showed significantly improved learning results for Lang 55 and Lang 3. Furthermore, it contributes to more efficient convergence. A learner using the ERC method reaches convergence after a much smaller number of grammar changes.

Appendix A Appendix A: Full word list of Lang 55 and Lang 3

The number 1 corresponds to primary stress, and the number 2 to secondary stress.

Table 4: Full list of words for Language 55

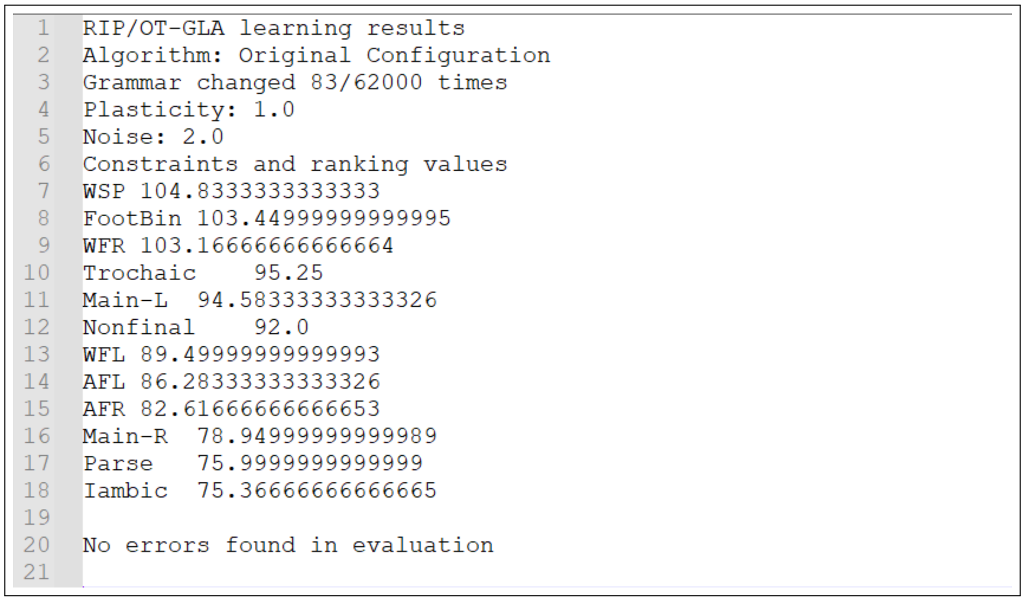
[L1 L]	[L H1 H2]	[H1 L H2 L]	[L1 L H2 L H2]	[H1 L L L2 L]	[H1 H2 L H2 H2]
[L H1]	[L1 L L2 L]	[H1 L H2 H2]	[L1 L H2 H2 L]	[H1 L L L H2]	[H1 H2 H2 L2 L]
[H1 L]	[L1 L L H2]	[H1 H2 L2 L]	[L1 L H2 H2 H2]	[H1 L L H2 L]	[H1 H2 H2 L H2]
[H1 H2]	[L1 L H2 L]	[H1 H2 L H2]	[L H1 L L2 L]	[H1 L L H2 H2]	[H1 H2 H2 H2 L]
[L L1 L]	[L1 L H2 H2]	[H1 H2 H2 L]	[L H1 L L H2]	[H1 L H2 L2 L]	[H1 H2 H2 H2 H2]
[L1 L H2]	[L H1 L2 L]	[H1 H2 H2 H2]	[L H1 L H2 L]	[H1 L H2 L H2]	[L1 L L L L2 L]
[L H1 L]	[L H1 L H2]	[L1 L L L2 L]	[L H1 L H2 H2]	[H1 L H2 H2 L]	[L1 L L L L2 L]
[H1 L2 L]	[L H1 H2 L]	[L1 L L L H2]	[L H1 H2 L2 L]	[H1 L H2 H2 H2]	
[H1 L H2]	[L H1 H2 H2]	[L1 L L H2 L]	[L H1 H2 L H2]	[H1 H2 L L2 L]	
[H1 H2 L]	[H1 L L2 L]	[L1 L L H2 H2]	[L H1 H2 H2 L]	[H1 H2 L L H2]	
[H1 H2 H2]	[H1 L L H2]	[L1 L H2 L2 L]	[L H1 H2 H2 H2]	[H1 H2 L H2 L]	

Table 5: Full list of words for Language 3

[L L1]	[H1 H2 H2]	[H1 L H2 L]	[L L1 H2 L H2]	[H1 L L2 L L2]	[H1 H2 L H2 H2]
[L H1]	[L L1 L L2]	[H1 L H2 H2]	[L L1 H2 H2 L]	[H1 L L2 L H2]	[H1 H2 H2 L L2]
[H1 L]	[L L1 L H2]	[H1 H2 L L2]	[L L1 H2 H2 H2]	[H1 L L2 H2 L]	[H1 H2 H2 L H2]
[H1 H2]	[L L1 H2 L]	[H1 H2 L H2]	[L H1 L L2 L]	[H1 L L2 H2 H2]	[H1 H2 H2 H2 L]
[L L1 L]	[L L1 H2 H2]	[H1 H2 H2 L]	[L H1 L L2 H2]	[H1 L H2 L L2]	[H1 H2 H2 H2 H2]
[L L1 H2]	[L H1 L L2]	[H1 H2 H2 H2]	[L H1 L H2 L]	[H1 L H2 L H2]	[L L1 L L2 L L2]
[L H1 L]	[L H1 L H2]	[L L1 L L2 L]	[L H1 L H2 H2]	[H1 L H2 H2 L]	[L L1 L L2 L L2 L]
[L H1 H2]	[L H1 H2 L]	[L L1 L L2 H2]	[L H1 H2 L L2]	[H1 L H2 H2 H2]	
[H1 L L2]	[L H1 H2 H2]	[L L1 L H2 L]	[L H1 H2 L H2]	[H1 H2 L L L2]	
[H1 L H2]	[H1 L L L2]	[L L1 L H2 H2]	[L H1 H2 H2 L]	[H1 H2 L L2 H2]	
[H1 H2 L]	[H1 L L2 H2]	[L L1 H2 L L2]	[L H1 H2 H2 H2]	[H1 H2 L H2 L]	

Appendix B   Examples of printed results at the end of a learning trial

Figure 8: Printed result of a successful learning trial of Lang 55 under the OT method

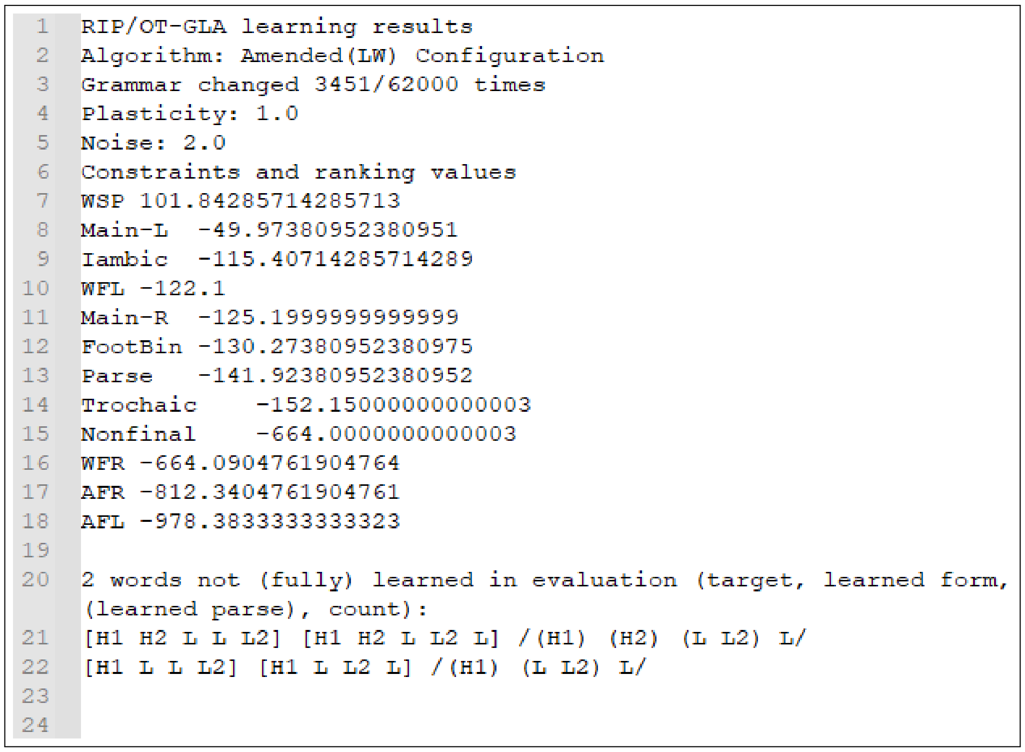


```
RIP/OT-GLA learning results
Algorithm: Original Configuration
Grammar changed 83/62000 times
Plasticity: 1.0
Noise: 2.0
Constraints and ranking values
WSP 104.8333333333333
FootBin 103.44999999999995
WFR 103.16666666666664
Trochaic    95.25
Main-L  94.58333333333326
Nonfinal    92.0
WFL 89.49999999999993
AFL 86.28333333333326
AFR 82.61666666666653
Main-R  78.94999999999989
Parse   75.9999999999999
Iambic  75.36666666666665

No errors found in evaluation

```

Figure 9: Printed result of a failed learning trial of Lang 3 under the ERC method



```
RIP/OT-GLA learning results
Algorithm: Amended(LW) Configuration
Grammar changed 3451/62000 times
Plasticity: 1.0
Noise: 2.0
Constraints and ranking values
WSP 101.84285714285713
Main-L  -49.97380952380951
Iambic  -115.40714285714289
WFL -122.1
Main-R  -125.1999999999999
FootBin -130.27380952380975
Parse   -141.92380952380952
Trochaic    -152.15000000000003
Nonfinal    -664.0000000000003
WFR -664.0904761904764
AFR -812.3404761904761
AFL -978.3833333333323

2 words not (fully) learned in evaluation (target, learned form, (learned parse), count):
[H1 H2 L L L2] [H1 H2 L L2 L] /(H1) (H2) (L L2) L/
[H1 L L L2] [H1 L L2 L] /(H1) (L L2) L/


```

References

Apoussidou, Diana. 2006. On-line learning of underlying forms. URL `http://roa.rutgers.edu/article/view/845`, available on the Rutgers Optimality Archive.

Apoussidou, Diana, and Paul Boersma. 2003. The learnability of Latin stress. *Proceedings of the Institute of Phonetic Sciences* 25:101–148.

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

Boersma, Paul, and Bruce Hayes. 2001. Empirical tests of the gradual learning algorithm. *Linguistic Inquiry* 32:41–86.

Boersma, Paul, and Joe Pater. 2008. Convergence properties of a gradual learning algorithm for harmonic grammar. *Ms.* .

Boersma, Paul, and David Weenink. 2022. Praat: doing phonetics by computer [Computer program]. URL `http://www.praat.org/`, version 6.2.12, retrieved 17 April 2022.

Hayes, Bruce. 1995. *Metrical Stress Theory: Principles and Case Studies*. University of Chicago Press.

Jarosz, Gaja. 2013. Learning with hidden structure in optimality theory and harmonic grammar: Beyond robust interpretive parsing. *Phonology* 30:27–71.

Kager, René. 2007. Feet and metrical stress. In *The cambridge handbook of phonology*, ed. Paul de Lacy, 195–227. Cambridge University Press.

Magri, Giorgio. 2012. Convergence of error-driven ranking algorithms. *Phonology* 29:213–269.

Magri, Giorgio, and Benjamin Storme. 2020. Calibration of constraint promotion does not help with learning variation in stochastic Optimality Theory. *Linguistic Inquiry* 51:97–123.

Pater, Joe. 2008. Gradual learning and convergence. *Linguistic Inquiry* 39:334–345.

Prince, Alan. 2002. Entailed ranking arguments. Ms. Available as ROA-500 from the Rutgers Optimality Archive.

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

Tesar, Bruce, and Paul Smolensky. 2000. *Learnability in optimality theory*. MIT Press.