Non-convergence in the GLA and variation in the CDA*

Joe Pater, University of Massachusetts, Amherst
October 24, 2005

Given a linguistic theory with constraints/parameters/rules of any generality, learners are faced with what Dresher (1999) calls the credit problem. In terms of Optimality Theory (OT: Prince and Smolensky 1993/2004), there is usually more than one constraint that favors an optimal form over any one of its competitors, and that can be ranked by the learner over the constraints preferring the competitor. One of the most attractive properties of the constraint demotion algorithm (CDA: Tesar and Smolensky 1998) is that it finds a ranking that correctly deals with the learning data (if one exists), without directly stipulating a solution to the credit problem (cf. Dresher 1999). In this paper, I show that an alternative algorithm for learnability in OT, the Gradual Learning Algorithm (GLA: Boersma 1997, 1998, Boersma and Hayes 2001), does not share this property: it fails to converge when presented with a set of data that has multiple interacting instantiations of the credit problem. The advantage of the GLA over the CDA is that it handles variation; in the second part of the paper, I sketch an approach to the learning of variation that makes use of the inconsistency detection properties of the CDA (Tesar 1998).

1. Non-convergence in the GLA

A piece of learning data in Optimality Theory consists of an input and the optimal output, or Winner, paired with another output candidate, or Loser (a mark-data pair or M-D pair). Constraints are annotated for whether they prefer the Winner or the Loser (this is an alternative to showing the violation marks incurred by each; see esp. Prince 2003). Using this notation, a simple instance of the credit problem would be as in (1), where two constraints (Con1 and Con3) favor the winner, and Con 2 favors the loser.

(1) *OT credit problem schematized*

	W ~ L	Con1	Con2	Con3
Input	$\text{Output}_W \sim \text{Output}_L$	W	L	W

The correct ranking must place Con1 or Con3 over Con2 – but which one? A decision can be made based on a further piece of data:

(2) *Credit problem resolved*

	W ~ L	Con1	Con2	Con3
In-1	$\text{Out-1}_W \sim \text{Out-1}_L$	W	L	W
In-2	$\text{Out-2}_W \sim \text{Out-2}_L$		W	L

The second M-D pair provides evidence that Con2 dominates Con3, so the first M-D pair must be dealt with by Con1 >> Con2, giving us the ranking Con1 >> Con 2 >> Con3.

* Thanks to Paul Boersma and Bruce Hayes for very helpful discussion, and to the participants in Ling 730, UMass Fall 2005, for feedback. The Praat files discussed here are posted on the internet as http://people.umass.edu/pater/WLW-5.txt and http://people.umass.edu/pater/WLW-5dist.txt.

A version of the CDA that deals with one piece of data at a time will find this ranking by iteratively demoting the loser preferring constraint in the current mark-data pair beneath the highest ranked winner preferring constraint (Tesar and Smolensky 1998). A version of the CDA that accumulates mark-data pairs (Tesar 1998 *et seq.*) will respond to (2) by first ranking Con1, as it is the only constraint that prefers only winners. This will eliminate the first M-D pair, allowing Con2 to be ranked in the next stratum, followed by Con3.

The GLA is cast within a stochastic version of OT that places constraints on a numeric scale (Boersma 1998). Each time the grammar is used to evaluate Input-Output pairings, the ranking values are perturbed slightly (by 'evaluation noise') before converting the values to a ranking order. This produces variation between the rankings of constraints whose values are close to one another. Learning in the GLA consists of changing the values on the basis of the learning data. In the version proposed by Boersma and Hayes (2001), the values of all constraints preferring the winner are raised, and the values of those preferring the loser are lowered, all by an equal amount termed the plasticity. Thus, on the basis of the first M-D pair in (2), the values of both Con1 and Con3 would be raised, and Con2 lowered. The second M-D pair would exert the opposite pressures on Con3 and Con2. Like most implementations of the CDA, the GLA is error-driven: mark-data pairs are created when the learner's grammar produces an Output that does not match the Output in the learning data (these are the Loser and Winner respectively). If Con1 reaches a value high enough above Con2 that errors become rare on Output-1_W, errors on Output-2_W, and associated learning, will place Con2 at a sufficient distance above Con3.

There is no formal proof that this procedure will succeed, but in practice, it does seem to consistently yield the right results on a case like that in (2). In (3), a more complex set of interacting credit problems is schematized.[1]

(3) *Multiple credit problems*

	W ~ L	Con1	Con2	Con3	Con4	Con5
In-1	Out-1_W ~ Out-1_L	W	L	W		
In-2	Out-2_W ~ Out-2_L		W	L	W	
In-3	Out-3_W ~ Out-3_L			W	L	W
In-4	Out-4_W ~ Out-4_L				W	L

In this learning scenario, each M-D pair resolves the credit problem in the pair above it. The fourth M-D pair requires Con4 >> Con5, so the third one must be dealt with by Con3 >> Con4, which entails a ranking of Con2 >> Con3 for the second M-D pair, which in turn forces Con1 >> Con2 for the first M-D pair. As a whole, these data require the ranking Con1 >> Con2 >> Con3 >> Con4 >> Con5.

As far as can be determined, the version of the GLA in Boersma and Hayes (2001) never converges when presented with this data set. I have tested this using the implementation of the GLA in Praat (v. 4.2, Boersma and Wernick 2004), with the standard settings for evaluation

[1] This seems to be the simplest case of WLW marks iterated in this pattern that will cause non-convergence; thanks to Bruce Hayes for helping to identify it.

noise and plasticity. The grammar file consisted of the tableaux in (4), which correspond to the candidate pairs in the M-D pairs in (3).

(4) *Constraint tableaux*

Input1	Con1	Con2	Con3	Con4	Con5
$Output1_W$		*			
$Output1_L$	*		*		

Input2	Con1	Con2	Con3	Con4	Con5
$Output2_W$			*		
$Output2_L$		*		*	

Input3	Con1	Con2	Con3	Con4	Con5
$Output3_W$				*	
$Output3_L$			*		*

Input4	Con1	Con2	Con3	Con4	Con5
$Output4_W$					*
$Output4_L$				*	

The learning data consisted of the winners distributed with equal probability:

(5) Input1 → $Output1_W$
Input2 → $Output2_W$
Input3 → $Output3_W$
Input4 → $Output4_W$

The Praat standard settings present the learner with 400000 Input-Output mappings, so the learner saw each of these about 100000 times. Here is a typical outcome:

(6)

Constraint	*Ranking Value*
Con3	12392.505
Con4	12392.208
Con1	12391.368
Con2	12391.023
Con5	12390.103

The constraints started at ranking values of 100. Because the GLA is not converging on a ranking, the values continue to increase throughout learning, reaching these unusually high points. The values are still very close to one another, and will produce variation for all of the Inputs. In a test run giving this grammar each Input 100000 times, with an evaluation noise setting of 2, the following distributions were produced:

(7) Input1 → $Output1_W$ 79372
Input1 → $Output1_L$ 20628

Input2 → Output2_W 55753
Input2 → Output2_L 44247

Input3 → Output3_W 57566
Input3 → Output3_L 42434

Input4 → Output4_W 77814
Input4 → Output4_L 22186

This example, constructed to contain multiple interacting credit problems, shows that the GLA, unlike the CDA, is not fully convergent. Learning scenarios that are constructed to mimic real-life situations might not contain all and only this distribution of constraint violations and winner-loser pairs, and might not lead to such a dramatic learning failure. However, it seems that interacting credit problems of this type are represented in varying degrees in realistically complicated situations, and that they do cause more localized instances of non-convergence, affecting a subset of the constraints.

2. Variation in the CDA

The GLA will succeed on the learning problem presented in the last section if it is run in "demotion only" mode, in which it only decreases the values of loser preferring constraints, and does not change the value of winner preferring constraints. However, as Boersma (1997, 1998) shows, this version of the GLA fails to converge in cases involving variation. Thus, no extant version of the GLA is convergent on both the present learning problem, and on variation. This raises the question of whether the CDA, which can solve the learning problem in section 1, can also cope with variation.[2]

Variation is when a single Input maps to two Outputs in the same environment, so that the choice between the Outputs cannot be made by any constraint. A simple example presented in terms of mark-data pairs is provided in (8). The Input /bat/ produced in isolation is surfaces as [bat] in one instance, and [ba] in the other.

(8) *Variation in Mark-Data pairs*

	W ~ L	NoCoda	Max
/bat/	bat ~ ba	L	W
/bat/	ba ~ bat	W	L

Tesar and Smolensky (1998) point out that if the CDA is presented with variation, it will be stuck in an endless process of iterative demotion. In this case /bat/ → [bat] will result in NoCoda being demoted beneath Max, and /bat/ → [ba] will lead to Max being placed beneath NoCoda.

[2] Bruce Hayes (p.c.) reports that Goldwater and Johnson's (2003) Maximum Entropy model does succeed on the learning problem in section 1. Adoption of this model, however, entails the rejection of the OT tenet of strict constraint domination, which greatly increases the power of the theory.

If the learner accumulates M-D pairs, however, the result is different: it will fail to find a ranking. In the set of M-D pairs in (8), neither NoCoda nor Max prefers only Winners, so they cannot be ranked. Tesar (1998) uses this property of inconsistency detection to discover mistakes in posited prosodifications, while Tesar *et al.* (2003) and Tesar and Prince (2004) and use it to find mistaken assumptions about underlying representations (see also McCarthy 2004 on UR learning, as well as Ota 2004, and Pater 2004, to appear, for applications to exceptions and the creation of lexically specific constraints).

This suggests a solution to the problem of variation: if the CDA detects inconsistency in the M-D pairs for a single Input, the learner establishes separate sub-rankings in the hierarchy (cf. Anttila 1997, 2002). In production-oriented parsing, any sub-ranking can be used, thus producing variation in the output of the grammar. In parsing of further learning data, the learner tries all sub-rankings, and if one succeeds, it does not produce an error.

To provide an example that fleshes this out slightly, let us assume that the language in (8) also exhibits consistent deletion of consonants from clusters. The M-D pairs would be as in (9).

(9) *Variation and consistency in Mark-Data pairs*

	W ~ L	*Complex	NoCoda	Max
/bat/	bat ~ ba		L	W
/bat/	ba ~ bat		W	L
/bla/	ba ~ bla	W		L

The CDA will first install *Complex. Under the proposal here, it will then establish two rankings for NoCoda and Max, as in (10):

(10) *Complex >> Max >> NoCoda
*Complex >> NoCoda >> Max

Any subsequent learning datum that contains either a coda or NoCoda-motivated deletion will be dealt with by one of the rankings in of Max and NoCoda, and will not produce further errors. In producing /bat/, or any other Input form with a potential coda, either ranking of these two constraints is possible, which will lead to variation between deletion and faithful parsing of the coda.

The GLA has as a further goal the modeling of the statistical distribution of output variants. This too seems plausibly accomplished under this modified version of the CDA. To do so, the learner could keep track of how often it succeeds in parsing the learning data with each of the sub-rankings. This could then be encoded in terms of a probabilistic preference for one of the rankings (cf. Hammond 2004).

It will not be trivial to make this proposal sufficiently explicit for it to be computationally implemented. The learner must know which constraint violations are incurred by each morpheme in a string in order to assess whether one of them is displaying inconsistent behavior (see also

Pater to appear). When the CDA stalls, there may be several constraints waiting to be installed; the modified CDA must be able to find the constraints involved in inconsistency (Alan Prince and Bruce Tesar, p.c.). Furthermore, even equipped with this information about the locus of inconsistency, the learner has the further task of sorting through the various possible causes, though variation may be relatively easy to spot (misprosodification and exceptionality, for example, may be more easily confounded).

3. Conclusions

There seems to be at present no OT learning algorithm that is fully convergent, and deals with variation in the learning data (cf. footnote 2). Here I have presented an example of non-convergent behavior in the GLA, and suggested a way that the CDA might be modified to deal with variation.

References

Anttila, Arto. 1997. Deriving variation from grammar. In F. Hinskens, R. van Hout and W. L. Wetzels (eds.) Variation, Change and Phonological Theory. Amsterdam, John Benjamins.

Anttila, Arto. 2002. Morphologically Conditioned Phonological Alternations. *Natural Language and Linguistic Theory* 20. 1-42.

Boersma, Paul. 1997. How We Learn Variation, Optionality, and Probability. Institute of Phonetic Sciences, University of Amsterdam, Proceedings 21. 43-58

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

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

Dresher, Bezalel Elan. 1999. Charting the Learning Path: Cues to Parameter Setting. *Linguistic Inquiry* 30: 27-67.

Goldwater, Sharon, and Mark Johnson. 2003. Learning OT Constraint Rankings Using a Maximum Entropy Model. *Proceedings of the Workshop on Variation within Optimality Theory*, Stockholm University.

Hammond, Michael. 2004. Gradience, Phonotactics, and the Lexicon in English Phonology. *International Journal of English Studies*. 1-24.

McCarthy, John. 2004. Taking a Free Ride in Morphophonemic Learning. Ms, University of Massachusetts, Amherst. (To appear in the *Catalan Journal of Linguistics*).

Ota, Mits. 2004 The learnability of the stratified phonological lexicon. *Journal of Japanese Linguistics* 20, 4.

Pater, Joe. 2004. Exceptions in Optimality Theory: Typology and Learnability. Presented at the Conference on Redefining Elicitation: Novel Data in Phonological Theory, New York University. (http://people.umass.edu/ pater/exceptions.pdf)

Pater, Joe. To appear.The Locus of Exceptionality: Morpheme-specific phonology as constraint indexation. In L. Bateman, M. O'Keefe, E, Reilly, and A. Werle (eds.), *University of Massachusetts Occasional Papers in Linguistics 32: Papers in Optimality Theory III*. Amherst: GLSA.

Prince, 2003. Arguing Optimality. In A. Carpenter, A. Coetzee, P. de Lacy (eds.) *University of Massachusetts Occasional Papers in Linguistics*: *Papers in Optimality Theory II*.

Prince, Alan, and Paul Smolensky. 1993/2004. *Optimality Theory: Constraint interaction in generative grammar*. Technical Report, Rutgers University and University of Colorado at Boulder, 1993. Revised version published by Blackwell, 2004.

Tesar, Bruce. 1998. Using the mutual inconsistency of structural descriptions to overcome ambiguity in language learning. In P. Tamanji and K. Kusumoto (eds.) Proceedings of the North East Linguistic Society 28. Amherst, MA: GLSA, University of Massachusetts. 469-483.

Tesar, Bruce, and Alan Prince. 2004. Using Phonotactics to Learn Phonological Alternations. In The Proceedings of CLS 39, Vol. II: The Panels.

Tesar, Bruce, Alderete, John, Horwood, Graham, Merchant, Nazarre, Nishitani, Koichi, and Prince, Alan. 2003. Surgery in language learning. In *Proceedings of the Twenty-Second West Coast Conference on Formal Linguistics*, ed. by G. Garding and M. Tsujimura, 477-490. Somerville, MA: Cascadilla Press.

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