Phonology **27** (2010) 77–117. © Cambridge University Press 2010
doi:10.1017/S0952675710000047

*Harmonic Grammar with linear programming: from linear systems to linguistic typology**

Christopher Potts
Stanford University

Joe Pater
Karen Jesney
Rajesh Bhatt
University of Massachusetts, Amherst

Michael Becker
Harvard University

Harmonic Grammar is a model of linguistic constraint interaction in which well-formedness is calculated in terms of the sum of weighted constraint violations. We show how linear programming algorithms can be used to determine whether there is a weighting for a set of constraints that fits a set of linguistic data. The associated software package OT-Help provides a practical tool for studying large and complex linguistic systems in the Harmonic Grammar framework and comparing the results with those of OT. We first describe the translation from harmonic grammars to systems solvable by linear programming algorithms. We then develop a Harmonic Grammar analysis of ATR harmony in Lango that is, we argue, superior to the existing OT and rule-based treatments. We further highlight the usefulness of OT-Help, and the analytic power of Harmonic Grammar, with a set of studies of the predictions Harmonic Grammar makes for phonological typology.

1 Introduction

We examine a model of grammar that is identical to the standard version of Optimality Theory (OT; Prince & Smolensky 2004), except that the

* Our thanks to Ash Asudeh, Tim Beechey, Maitine Bergonioux, Paul Boersma, John Colby, Kathryn Flack, Edward Flemming, Bob Frank, John Goldsmith, Maria Gouskova, Bruce Hayes, René Kager, Shigeto Kawahara, John Kingston, John McCarthy, Andrew McKenzie, Ramgopal Mettu, Alan Prince, Kathryn Pruitt, Jason Riggle, Jim Smith and Paul Smolensky, and other participants in conferences and courses where this material was presented. This material is based upon work supported by the National Science Foundation under Grant BCS-0813829 to Pater. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

optimal input–output mapping is defined in terms of weighted rather than ranked constraints, as in Harmonic Grammar (HG; Legendre *et al.* 1990a, b; see Smolensky & Legendre 2006 and Pater 2009b for overviews of subsequent research). We introduce a method for translating learning problems in this version of HG into linear models that can be solved using standard algorithms from linear programming. The implementation of this method facilitates the use of HG for linguistic research.

The linear programming model returns either a set of weights that correctly prefers all of the intended optimal candidates over their competitors or a verdict of 'infeasible' when no weighting of the given constraints prefers the indicated optima. Thus we provide for HG the equivalent of what the Recursive Constraint Demotion algorithm (Tesar & Smolensky 1998b) provides for OT: an algorithm that returns an analysis for a given data set with a given constraint set, and that also detects when no such analysis exists. In addition, we present OT-Help (Becker & Pater 2007, Becker *et al.* 2007), a graphically based program that can take learning data formatted according to the standards defined for the software package OTSoft (Hayes *et al.* 2003) and solve them using our linear programming approach (and with Recursive Constraint Demotion).[1] The public availability of OT-Help will help research on weighted constraint interaction to build on results already obtained in the OT framework.

We start by discussing the model of HG we adopt and its relationship to its better-known sibling OT (§2). §3 states the central learning problem of the paper. We then describe our procedure for turning HG learning problems into linear programming models (§4). §5 develops an HG analysis of an intricate pattern of ATR harmony in Lango. The analysis depends crucially on the kind of cumulative constraint interaction that HG allows, but that is impossible in standard OT. We argue that the HG approach is superior to Archangeli & Pulleyblank (1994)'s rule-based analysis and Smolensky (2006)'s constraint-conjunction approach. Finally, §6 is a discussion of typology in HG, with special emphasis on using large computational simulations to explore how OT and HG differ. That discussion deepens our comparison with OT, and it highlights the usefulness of using efficient linear programming algorithms to solve linguistic systems. We show that comparisons between OT and HG depend on the contents of the constraint sets employed in each framework, and that the greater power of HG can in some cases lead, perhaps surprisingly, to more restrictive typological predictions.

2 Overview of Harmonic Grammar

In an optimisation-based theory of grammar, a set of constraints chooses the optimal structures from a set of CANDIDATE structures. In this paper,

[1] In addition, the popular open-source software package Praat (Boersma & Weenink 2009) now offers an HG solver designed using the method we introduce here.

candidates are pairs $\langle In, Out\rangle$, consisting of an input structure *In* and an output structure *Out*. In HG, optimality is defined in terms of a harmony function that associates each candidate with the weighted sum of its violations for the given constraint set. The weighted sum takes each constraint's violation count and multiplies it by that constraint's weight, and sums the results.

(1) **Definition 1** (harmony function)

Let $\mathbf{C} = \{C_1 \ldots C_n\}$ be a set of constraints, and let W be a total function from $\mathbf{C}$ into positive real numbers. Then the HARMONY of a candidate A is given by:

$$\mathcal{H}_{\mathbf{C},W}(A) = \sum_{i=1}^{n} W(C_i) \cdot C_i(A)$$

We insist on only positive weights. While there is no technical problem with allowing a mix of negative and positive weights into HG, the consequences for linguistic analysis would be serious. For example, a negative weight could turn a penalty (violation count) into a benefit. For additional discussion of this issue, see Prince (2003), Boersma & Pater (2008: §3.5) and Pater (2009b: §2.1).

The constraints themselves are functions from candidates into integers. We interpret $C(A) = -4$ to mean that candidate A incurs four violations of constraint C. We also allow positive values: $C(A) = 4$ thus means that A satisfies constraint C four times. In this paper, we use only constraint violations (negative numbers), but the approach we present is not limited in this way.

The optimal candidates have the highest harmony scores in their candidate sets. Since we represent violations with negative natural numbers, and weights are positive, an optimum will have the negative score closest to zero, which can be thought of as the smallest penalty. As in OT, this competition is limited to candidates that share a single input structure. In anticipation of the discussion in §4, we make this more precise by first defining the notion of a TABLEAU, the basic domain over which competitions are defined.

(2) **Definition 2** (tableaux)

A TABLEAU is a structure $(\mathbf{A}_{In}, \mathbf{C})$, where $\mathbf{A}_{In}$ is a (possibly infinite) set of candidates sharing the input *In*, and $\mathbf{C}$ is a (finite) constraint set.

We can then define optimality in terms of individual tableaux: the optimum is a candidate that has greater harmony than any of the other members of its candidate set.

(3) **Definition 3** (optimality)

Let $T = (\mathbf{A}_{In}, \mathbf{C})$ be a tableau, and let W be a weighting function for $\mathbf{C}$. A candidate $A = \langle In, Out\rangle \in \mathbf{A}_{In}$ is optimal iff $\mathcal{H}_{\mathbf{C},W}(A) > \mathcal{H}_{\mathbf{C},W}(A')$ for every $A' \in (\mathbf{A}_{In} - \{A\})$.

The use of a strict inequality rules out ties for optimality, and brings our HG model closer to the standard version of OT, whose totally ordered constraint set also typically selects a unique optimum (if the constraint set is large enough). Languages with tied optima are not of particular interest, since the resulting variation is unlikely to match actual language variation (see the discussion below of existing theories of stochastic OT, which either render ties vanishingly improbable, as in Noisy HG, or completely eliminate the notion of an optimum, defining instead a probability distribution over candidates, as in Maximum Entropy grammar).

Goldsmith (1991: 259) proposes to model phonological interactions using weighted constraints; he describes an account in which constraint violations can involve variable costs, which encode relative strength and determine relative well-formedness. Goldsmith (1990: §6.5)'s discussion of violability and cost accumulation contains clear antecedents of these ideas; see also Goldsmith (1993, 1999).

Prince & Smolensky (2004: 236) also discuss a version of OT that uses weighted sums to define optimality. Our formulation follows that of Keller (2000, 2006) and Legendre *et al.* (2006), though it differs from Legendre *et al.*'s in demanding that an optimum in a candidate set be unique, which is enforced by using a strict inequality (the harmony of an optimum is greater than its competitors). This is a simplifying assumption that allows for easier comparison with the typological predictions of OT. Example (4) is a typical representation of a tableau for HG. The single shared input is given in the upper left, with candidate outputs below it and their violation scores given in tabular format. The representation is like those used for OT, but without ranking being signified by the left-to-right order; it also adds a weighting vector in the topmost row and the harmony scores for each candidate in the rightmost column.

(4) *A weighted constraint tableau*

weight	2	1	$\mathcal{H}$
Input	C_1	C_2	
☞ a. Output_a	0	−1	−1
b. Output_b	−1	0	−2

By (3), Output_a is chosen as the optimal output for Input. Optimal candidates are marked with the pointing finger.

We emphasise that our version of HG, as characterised by (3), is, like OT, an optimisation system. Our HG grammars do not impose a single numerical cut-off on well-formedness, but instead choose the best outcome for each input. This point is vital to understanding how the systems work, but it is easily overlooked. We therefore pause to illustrate this with a brief example modelled on one discussed by Prince & Smolensky (1997: 1606; for additional discussion, see Pater 2009b). We assume that it is typologically implausible that we will find a natural language in which a single coda is tolerated in a word but a second coda is deleted. Such a language would map the input /ban/ faithfully to [ban], but would map input /bantan/ to [ba.tan] or [ban.ta]. Such patterns are unattested, arguably for fundamental reasons about how natural languages work, so we would like our theory to rule them out. In OT, it can be shown that this pattern would require contradictory rankings: NoCoda would have to outrank, and be outranked by, Max, which is impossible. HG delivers exactly the same result. To make deletion of one of two potential codas optimal, as in (5a), NoCoda must have a weight greater than Max. To make preservation of a single potential coda optimal, as in (5b), Max must have a greater weight than NoCoda. (We use specific weights to illustrate how the calculations work.)

(5) a.

weight	2	1	$\mathcal{H}$
/bantan/	NoCoda	Max	
i. ban.tan	−2	0	−4
☞ ii. ba.tan	−1	−1	−3

b.

weight	1	2	$\mathcal{H}$
/ban/	NoCoda	Max	
☞ i. ban	−1	0	−1
ii. ba	0	−1	−2

The contradictory weighting conditions for (5a) and (5b) can be represented more generally, as in (6a) and (6b) respectively. These statements are the HG analogues of the contradictory pair of ranking statements we would require in OT.

(6) a. W(NoCoda) > W(Max) b. W(NoCoda) < W(Max)

What's more, if we include complete candidate sets for the evaluation, then assigning greater weight to NoCoda selects the mapping /bantan/ → [ba.ta] as optimal, whereas assigning greater weight to Max selects the mapping /bantan/ → [ban.tan] as optimal, just like the two possible total rankings of these constraints in OT.

Importantly, if we were to impose a numerical cut-off on well-formedness, then we could rule out /bantan/ → [bantan] but allow /batan/ → [batan] (with, for example, W(NoCoda) = 2, and the cut-off above 2 and below 4). However, our version of HG does not impose numerical cut-offs (for versions of HG that do generate this sort of pattern, see Jäger 2007 and Albright *et al.* 2008).

As this example illustrates, the grammatical apparatus is designed to model entire languages, not just individual mappings from input to optimal output. Thus, we deal primarily with TABLEAU SETS, which are sets of tableaux that share a single set of constraints but have different inputs.

(7) **Definition 4** (tableau sets)

A tableau set is a pair ($\mathbf{T}$, $\mathbf{C}$) in which $\mathbf{T}$ is a set of tableaux such that if $T = (\mathbf{A}_{In}, \mathbf{C}') \in \mathbf{T}$ and $T' = (\mathbf{A}'_{In'}, \mathbf{C}'') \in \mathbf{T}$ and $T \neq T'$, then $In \neq In'$ and $\mathbf{C} = \mathbf{C}' = \mathbf{C}''$.

Given a tableau set (**T**, **C**), a weighting function W determines a language by selecting the optimal candidate, if there is one, from each tableau $T \in \mathbf{T}$. Since our (3) uses a strict inequality, some tableaux could theoretically lack optimal candidates. We note again that by using a strict inequality, our definition involves a simplification. Versions of the theory that are designed to handle variation between optima across instances of evaluation do not make this simplifying move (see e.g. Boersma & Pater 2008).

HG is of interest not only because it provides a novel framework for linguistic analysis, but also because its linear model is computationally attractive. HG was originally proposed in the context of a connectionist framework. OT ranking has so far resisted such an implementation (Prince & Smolensky 2004: 236, Legendre *et al.* 2006: 347). Beyond connectionism, HG can draw on the well-developed models for learning and processing with linear systems in general, and the basic apparatus can be used in a number of different ways. For example, a currently popular elaboration of the core HG framework we explore here is the probabilistic model of grammar proposed by Johnson (2002), and subsequently applied to phonology by Goldwater & Johnson (2003), Wilson (2006), Jäger (2007) and Hayes *et al.* (2008). In this log-linear model of grammar, usually referred to as Maximum Entropy (MaxEnt) grammar, the probability of a candidate is proportional to the exponential of its harmony, calculated as in (1) above. By assigning a probability distribution to candidates, a MaxEnt grammar can deal with the 'free variation' that is captured in OT as probabilistic variation in the ranking of constraints (see Coetzee & Pater, in press for an overview of OT and HG models of variation). As the above-cited papers emphasise, MaxEnt grammar is particularly appealing in that it has associated provably convergent learning algorithms, unlike extant approaches to variation in OT.

Stochastic versions of HG like MaxEnt grammar and Boersma & Pater (2008)'s Noisy HG can subsume our categorical model as a special case in which all candidates have probabilities approaching 1 or 0. What, then, is the interest of our more restrictive approach? We see it as a useful idealisation that facilitates analysis of individual languages and typological study. Even when the analyst's ultimate aim is to provide a stochastic HG account of a case involving variation, it can be useful to first develop a categorical analysis of a subset of the data.

Despite its attractive properties, HG has been little used in analysing the patterns of constraint interaction seen in the grammars of the world's languages. One possible reason for the relative neglect is that researchers may have assumed that HG would clearly produce unwanted typological results (Prince & Smolensky 2004: 233). In related work, Pater (2009b) argues that this assumption should be re-examined. By studying a categorical version of HG that differs so minimally from OT, it becomes possible to gain a clearer understanding of the difference between a theory of grammar that has ranked constraints and one that has weighting. Both the Lango ATR-harmony analysis of §5 and the typological investigations of §6 focus on uncovering these differences.

Another likely reason that HG is relatively understudied is that it can be difficult to calculate by hand a weighting for a set of constraints that will correctly prefer the observed output forms over their competitors. Furthermore, in doing linguistic analysis, we are often interested in showing that a particular set of outputs can never coexist in a single language, that is, in showing that a theory is sufficiently restrictive. Establishing that none of the infinitely many possible weightings of a set of constraints can pick a set of outputs as optimal may seem to be an insurmountable challenge. These problems are the motivation for our development of a translation from HG learning to linear programming solving, and for the implementation of this procedure in OT-Help.

3 Our Harmonic Grammar learning problem

The learning problem that we address, in this paper and with OT-Help, is defined in (8).

(8) Let $(\mathbf{T}, \mathbf{C})$ be a tableau set, and assume that each tableau $T = (\mathbf{A}_{In}, \mathbf{C}) \in \mathbf{T}$ is finite and contains exactly one designated intended winning candidate $o \in \mathbf{A}_{In}$. Let O be the set of all such intended winners. Is there a weighting of the constraints in $\mathbf{C}$ that defines all and only the forms in O as optimal (definition 3)? If so, what is an example of such a weighting?

This is analogous to a well-studied problem in OT (Tesar & Smolensky 1998b, Prince 2002, Prince & Smolensky 2004): given a set of grammatical forms and a set of constraints **C**, is there a ranking of the constraints in **C** that determines all and only the grammatical forms to be optimal?

Our approach to the problem is categorical: for a given subset A of the full set of candidates, the algorithm either finds a harmonic grammar that specifies all and only the members of A as optimal, or else it answers that there is no such harmonic grammar. This is by no means the only approach one could take to (8) and related questions. As we mentioned earlier, there are many useful perspectives, including those that allow for approximate learning, those that model learning in noise and so forth. Our particular approach to answering (8) is ideally suited to examining typological predictions of the sort discussed in §6 below.

We do not in this paper address the issue of candidate generation, focusing instead on the twin problems of evaluation and typological prediction. Like the OT implementations in OTSoft (Hayes *et al.* 2003) and Praat (Boersma & Weenink 2009), and the HG implementations in Praat and those discussed in the MaxEnt literature cited above, we take the candidates as given. This introduces the risk of spurious conclusions based on non-representative candidate sets, but we see no satisfactory way around the problem at present. Riggle (2004a, b) shows how to generate the full set of candidates which are optimal under some ranking of the constraints, but that holds only if, among other things, all the constraints have finite-state implementations. The result carries over to HG. However, it would be a mistake for us to prematurely limit HG to this set of constraints in this way. This is not a limitation that the OT community has made, and we know of no reason to assume that HG makes this more pressing than it is in other approaches.

Although HG does not impose finiteness limitations on its candidate sets, (8) restricts attention to the finite case, in recognition of the fact that OT-Help can deal only with finite systems. There are linear programming methods for dealing with situations in which, in present terms, the candidate set is infinite but the constraint set is finite; López & Still (2007) is an overview.[2] However, exploring such algorithms is outside the bounds of this paper. In addition, we suspect that the proper approach here is not to explicitly allow infinite candidate sets, but rather to take a constructive approach to exploring the space of potential winners, as in Harmonic Serialism (McCarthy 2007, 2009, Pater, to appear).[3]

It is typically fairly easy to answer question (8) for small systems like (9). Here we follow Tesar & Smolensky (1998b) in referring to the desired optimal form as the 'winner', and the desired suboptimal candidates as the 'losers'.

(9)

weight				$\mathcal{H}$
Input	C_1	C_2	C_3	
a. Winner	−4	0	−4	
b. Loser	0	−2	0	

For (9), we can swiftly reason to the conclusion that a weighting ⟨1, 4.1, 1⟩ suffices. That is, we can use a weighting function W such that $W(C_1)=1$, $W(C_2)=4.1$ and $W(C_3)=1$. This gives ⟨Input, Winner⟩ a total weighted violation count of −8 and ⟨Input, Loser⟩ a total weighted violation count of −8.2. And it is easy to see that many other weightings work as well. But it quickly becomes challenging to reason in this way. Hand calculations are prohibitively time-consuming even for modestly sized systems. This is where linear programming methods become so valuable. They can answer question (8) quickly for even very large and complex systems. We turn now to the task of showing how to apply such algorithms to these data.

4 From linguistic data to linear systems

In this section, we build a bridge from linguistics into the domain of linear programming algorithms. In doing this, we make powerful and efficient

[2] We thank an anonymous reviewer for bringing this work to our attention.

[3] The next version of OT-Help will implement Harmonic Serialism, in which generation and evaluation are combined.

tools available to the linguist wishing to grapple with large, complex data sets. Our description closely follows the algorithm we employ in OT-Help, which incorporates the stand-alone HG solver HaLP (Potts *et al.* 2007), which has a web interface that allows users to upload their own data files and which displays its results in HTML. Our discussion proceeds by way of the simple tableau set in (10).

(10)

Input_1	C_1	C_2
a. Winner_1	0	−2
b. Loser_1	−6	0

Input_2	C_1	C_2
a. Winner_2	−1	0
b. Loser_2	0	−1

In OT, these two winner–loser pairs are inconsistent, since Winner_1 requires $C_1 \gg C_2$, and Winner_2 requires $C_2 \gg C_1$. Our primary task is to determine whether the same is true in HG, or whether there is a weighting vector $\langle w_1, w_2 \rangle$ that selects $\langle \text{Input}_1, \text{Winner}_1 \rangle$ and $\langle \text{Input}_2, \text{Winner}_2 \rangle$ as optimal.

4.1 Equations in the linear system

We first convert the weighting conditions into linear inequalities. For each winner–loser pair, we want an inequality that guarantees that the winner has greater harmony than the loser, as in (3). Weighting conditions like those in (6) are useful for getting a handle on the problem analytically. For the tableau depicted on the left in (10), the weighting condition is (11a), and for the tableau on the right in (10), the weighting condition is (11b).

(11) a. $(0 \cdot W(C_1)) + (-2 \cdot W(C_2)) > (-6 \cdot W(C_1)) + (0 \cdot W(C_2))$
b. $(-1 \cdot W(C_1)) + (0 \cdot W(C_2)) > (0 \cdot W(C_1)) + (-1 \cdot W(C_2))$

For the numerical optimisations to follow, we make extensive use of the following notation.

(12) a. $0w_1 + -2w_2 > -6w_1 + 0w_2 \Rightarrow 6w_1 + -2w_2 > 0$
b. $-1w_1 + 0w_2 > 0w_1 + -1w_2 \Rightarrow -1w_1 + 1w_2 > 0$

The w_i variables are the weights assigned by the weighting function W to these constraints. Inequality (12a) expresses the requirement that the Winner_1 output is favoured by the weighting over the Loser_1 output, and (12b) expresses the requirement that the Winner_2 output is favoured by the weighting over the Loser_2 output. These inequalities are the HG equivalents of OT's Elementary Ranking Conditions (Prince 2002). They can be directly calculated from a winner–loser pair by subtracting the loser's score on each constraint from that of the winner.

Given a tableau set (**T**, **C**), we translate each winner–loser pair in each tableau in **T** into an inequality statement like the above. A weighting answers the learning problem in (8) for (**T**, **C**) if and only if it satisfies all of these inequality statements simultaneously.

4.2 The objective function

All and only the vectors $\langle w_1, w_2 \rangle$ satisfying the inequalities in (12) are solutions to the learning problem (8) for (10). The vectors $\langle 1, 2 \rangle$ and $\langle 2, 3 \rangle$ suffice, as do an infinite number of others.

The structure of linear programming problems gives us an analytically useful way of selecting from the infinitude of possible solutions to a problem like this. The crucial notion is that of an OBJECTIVE FUNCTION. Throughout this paper, we work with very simple objective functions: just those that seek to minimise the sum of all the weights. Thus, for the two-constraint tableau set (10), the objective function is (13).

(13) minimise $\quad 1w_1 + 1w_2$

More generally, if there are n constraints, we seek to minimise the sum of all the weights w_i for $1 \leq i \leq n$, subject to the full set of inequalities for the system.

However, we have now run into a problem: our optimisation problem is undefined (Chvátal 1983: 43). The vector $\langle 1, 2 \rangle$ is not a minimal feasible solution, and neither are $\langle 1, 1.5 \rangle$, $\langle 1, 1.1 \rangle$, $\langle 1, 1.0001 \rangle$, etc. Each is better than the previous one according to (13); there is no minimal solution. Thus we can never satisfy (13); whatever solution we find can always be improved upon. The problem can be traced to our use of strict inequalities. In stating the problem this way, we are effectively stating a problem of the form 'find the smallest x such that $x > 0$', which is also ill-defined.

It won't do to simply change '$>$' to '$\geq$', because that would insist only that the winner be at least as good as the losers, whereas our version of HG demands that the winner be strictly better. Thus, to address this problem, we solve for a special constant a. It can be arbitrarily small, as long as it is above 0. It allows us to have regular inequalities without compromising our goal of having the winner win (not tie). This is equivalent to adding the amount a to the weighted sum of the loser's constraint violations. The value of a defines a margin of separation: the smallest harmony difference between an optimum and its nearest competitor. (Such margins of separation are important for the Perceptron convergence proof; see Boersma & Pater 2008 for an application to HG.)

Our use of the margin of separation a renders certain systems infeasible that would otherwise be feasible. These are the systems in which a winner can at best tie its losing competitors. We want these systems to be infeasible, because we want the winners to be strictly better. But one might wonder whether certain choices of a could rule out systems that we want to judge feasible. For instance, what happens if a is set to be very large? Could this incorrectly rule out a feasible analysis?

The answer is no. We assume that there is no maximal weighting for any constraint, and none of our systems contain the conditions that would impose such a ceiling for particular cases. Thus, assume that the chosen constant is a, and assume also that there is a weighting W for which one of the inequality statements sums to a constant d that is smaller than a. Then we simply find a linear rescaling of W that respects our choice of a rather than d. This rescaling could result in infeasibility only if there were a maximal value for some weight. But we assume that there are no such maxima.

4.3 Blocking zero weights

The next question we address is whether to allow 0 weights. A weighting of 0 is equivalent to cancelling out violation marks. To prevent such cancellation, we can impose additional conditions, over and above those given to us directly by the weighting conditions: for each constraint C_i, we can add the inequality $w_i \geq b$, for some positive constant b. Once again, because we impose no maxima, excluding this subregion does not yield spurious verdicts of infeasibility.

It is worth exploring briefly what happens if we remove the extra non-0 restrictions (if we set the minimal weight b to 0). In such systems, some constraint violations can be cancelled out when weighted, via multiplication by 0. This cancellation occurs when a given constraint is inactive for the data in question, i.e. when it is not required in order to achieve the intended result. For example, our current model returns $\langle 1, 1, 1 \rangle$ as a feasible solution for the small system in (14) (assuming that we set the margin of separation a to 1 and the minimal weight b to 1).

(14)

weight	1	1	1	$\mathcal{H}$
Input	C_1	C_2	C_3	
a. Winner	0	−1	0	−1
b. Loser	−1	0	−1	−2

In this solution, C_1 and C_3 GANG UP on C_2: with this weighting, neither suffices by itself to beat the loser, but their combined weighted scores achieve the result. However, if we do not ensure that all weights are at least b, then the minimal solutions for these data are $\langle 1, 0, 0 \rangle$ and $\langle 0, 0, 1 \rangle$, with either of C_1 or C_3 decisive and the other two constraints inactive. As in this example, imposing a greater than 0 minimum on weights tends to result in solutions that make use of gang effects, while choosing a 0 minimum tends to find solutions that make use of a smaller number of constraints. Exploring the differences between these solutions (as is possible in OT-Help) may help an analyst better understand the nature of the constraint interactions in a system.

$$
\begin{array}{ll}
\text{minimise} & 1w_1 + 1w_2 \\
\text{subject to} & 6w_1 - 2w_2 \geq 1 \\
 & -1w_1 + 1w_2 \geq 1 \\
 & 1w_1 \qquad\quad \geq 1 \\
 & \qquad\quad 1w_2 \geq 1
\end{array}
$$

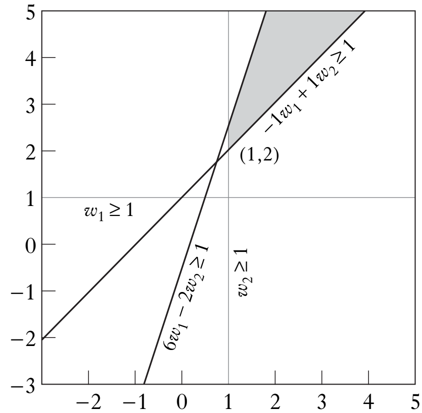

Figure 1

Translation and graph of (10), with the feasible region shaded.

4.4 The final form of the system

The linear system derived from (10) using the above procedure is given in Fig. 1, along with a geometric representation. To provide a concrete solution and a visualisation, we've set the margin of separation *a* to 1 and the minimal weight *b* to 1.[4] The optimal weighting here is $w_1 = 1$ and $w_2 = 2$. The current version of OT-Help accepts OTSoft files as input, converts them into tableau sets, translates them using the above procedure and then solves them with the simplex algorithm, the oldest and perhaps most widely deployed linear programming algorithm (Dantzig 1982, Chvátal 1983, Bazaraa *et al*. 2005).

4.5 Further remarks on the translation

Before putting these technical concepts to work solving linguistic problems, we would like to pause briefly to use graphical depictions like the one in Fig. 1 to clarify and further explore some of the decisions we made in translating from tableau sets to linear systems. Because each linguistic constraint corresponds to a dimension, we are limited to two-constraint systems when visualising, but the technique can nonetheless be illuminating.

4.5.1 *Infeasibility detection.* The graphical perspective immediately makes it clear why some linguistic systems are predicted to be impossible: they have empty feasible regions. Our simple NoCoda/Max example

[4] This is the default for OT-Help. An advantage of this is that it often returns integer-valued weights, which are helpful for studying and comparing systems.

$$\begin{array}{ll} \text{minimise} & 1w_1 + 1w_2 \\ \text{subject to} & -1w_1 + 1w_2 \geq 1 \\ & 1w_1 - 1w_2 \geq 1 \\ & 1w_1 \qquad\quad \geq 1 \\ & \qquad\quad 1w_2 \geq 1 \end{array}$$

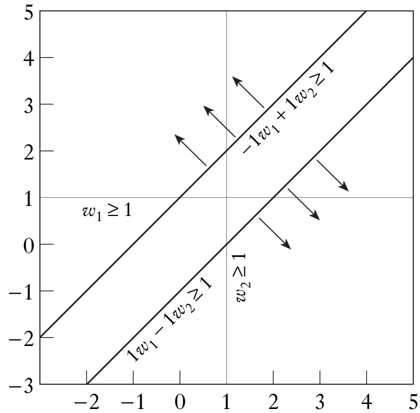

Figure 2

The linear view of tableau set (15). The intersection of all the areas picked out by the inequalities is empty, which is just to say that no grammar picks out the set of specified winners.

from §2 provides a good case study. Our goal there was to show that HG, like OT, predicts that it is impossible for a single language to allow a /ban/ to surface faithfully as [ban], but for it to penalise just one of the codas in /bantan/, thereby allowing something like [ba.tan] to surface. Here is a tableau set seeking to specify such a language.

(15)

/bantan/	NoCoda	Max
a. ban.tan	−2	0
☞ b. ba.tan	−1	−1

/ban/	NoCoda	Max
☞ a. ban	−1	0
b. ba	0	−1

In Fig. 2, we have transformed this tableau set into a linear system and plotted it. The arrows indicate which region the two main inequalities pick out. There is no area common to both of them, which is just to say that the feasible region is empty.

4.5.2 *Margins of separation.* We asserted in §4.2 that the precise value of *a* does not matter for addressing the fundamental learning problem (8). Figure 3 helps bring out why this is so. This figure differs minimally from the one in Fig. 1, in that the value of *a* here is 3 rather than 1. This narrows the bottom of the feasible region, and, in turn, changes the minimal solution, from ⟨1, 2⟩ to ⟨2¼, 5¼⟩, but the important structure of the system is unchanged.

One's choice of the margin of separation *a* can have consequences for how the solution generalises to unseen data, that is, to tableaux that are not included in the learning data. Suppose, for example, that we evaluate the

$$\begin{array}{ll} \text{minimise} & 1w_1 + 1w_2 \\ \text{subject to} & 6w_1 - 2w_2 \geq 3 \\ & -1w_1 + 1w_2 \geq 3 \\ & 1w_1 \geq 1 \\ & 1w_2 \geq 1 \end{array}$$

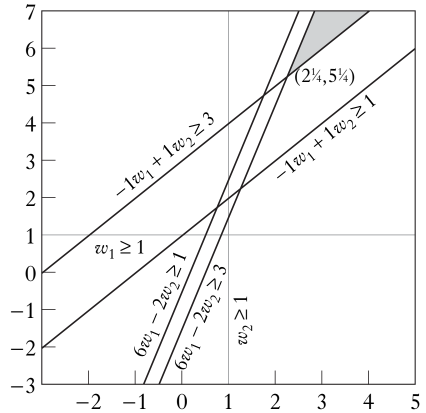

Figure 3

The system in Fig. 1, but with the value of *a* set to 3, rather than 1. The feasible region has narrowed at the bottom, and the solution is different, but the basic structure remains the same.

candidates in the following new tableau, using the weights found with each of the two values of *a* above.

(16)

Input_3	C_1	C_2
a. Output_{3a}	0	−4
b. Output_{3b}	−9	0

With $a = 3$, the optimal weighting vector is $\langle 2\frac{1}{4}, 5\frac{1}{4} \rangle$, which favours Output_{3a}. With $a = 1$, the optimal weighting vector is $\langle 1, 2 \rangle$, which favours Output_{3b}.

4.5.3 *Stopping short of optimization.* In discussing the objective function (§4.2), we emphasised finding minimal solutions. While knowing which is the minimal solution can be illuminating, it goes beyond the learning question (8), which simply asks whether there is a feasible solution at all. Our approach can be simplified slightly to address a version of this more basic question, with a resulting gain in efficiency. To see this, we need to say a bit more about how the simplex algorithm works.[5]

The simplex algorithm begins by setting all the weights to 0 and then pivoting around the edge of the feasible region until it hits the optimal

[5] We stay at a relatively informal level here, since full descriptions of the simplex algorithm invariably run to dozens of pages and involve making a variety of specific assumptions about data structures. Chvátal (1983) presents a variety of different formulations, Cormen *et al.* (2001: 29) give an accessible algebraic implementation in pseudocode and Bazaraa *et al.* (2005) is an advanced textbook devoted to the simplex algorithm as well as its newer, theoretically more efficient alternatives.

$$
\begin{array}{lrcr}
\text{minimise} & -1w_1 + 1w_2 & & \\
\text{subject to} & -4w_1 + 1w_2 & \geq & -8 \\
& -2w_1 - 1w_2 & \geq & -10 \\
& 5w_1 - 2w_2 & \geq & -2 \\
& \text{all } w_i & \geq & 0
\end{array}
$$

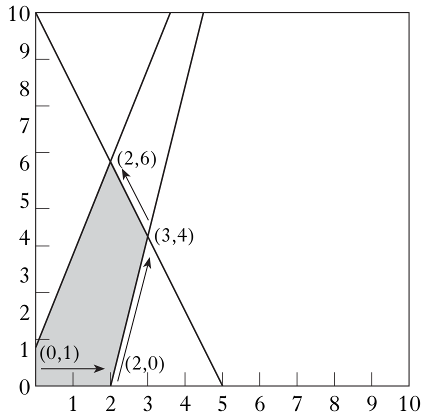

Figure 4

The simplex algorithm begins at the all-0s solution (the origin), and then pivots around the edge of the feasible region until it finds the vector that does best by the objective function.

solution according to the objective function. Figure 4 illustrates for one of the basic two-variable systems discussed by Cormen *et al.* (2001: 773). The arrows show one direction that the simplex might take; which direction it travels depends on low-level implementation decisions.

For this problem, the all-0s solution is inside the feasible region, so it provides a starting point. However, for all the systems arrived at via the conversion method of §4, setting all the weights to 0 results in an infeasible solution. For this reason, our solver always goes through two PHASES. In phase one, it constructs from the initial system an AUXILIARY SYSTEM for which the all-0s solution is feasible and uses this system to move into the feasible region of the initial problem (ending phase one). In Fig. 1, this auxiliary program takes us from the origin of the graph to the point $\langle 1, 2\tfrac{1}{2}\rangle$, which is a feasible solution. The phase two optimisation then brings us down to $\langle 1, 2\rangle$, which minimises the objective function.

The auxiliary program also provides us with a means for detecting infeasibility. One of the central pieces of this auxiliary program is a new artificial variable, w_0. After we have solved the auxiliary program, we check the value of this variable. If its value is 0, then we can safely remove it and, after a few additional adjustments, we have a feasible solution to the original problem. If its value is not 0, however, then it is crucial to our finding a solution in the first place, thereby indicating that the initial problem has no solutions. This is the source of the verdict of 'infeasible' – the linguist's cue that the grammar cannot deliver the desired set of optimal candidates.

Thus the question of whether there is a feasible weighting is answered during phase one of the simplex, with phase two devoted to potential improvements with regard to the objective function. If such improvements are not of interest, then we can stop at the end of phase one.

5 Lango ATR harmony in HG

We now turn to linguistic analysis using HG, and our linear programming method as implemented in OT-Help. A key argument for OT's violable constraints is their ability to reduce complex language-specific patterns to more general, plausibly universal principles. For example, Prince & Smolensky (2004: §4) show that a complex pattern of stress in the dialect of Hindi described by Kelkar (1968) can be reduced to the interaction of three general constraints. This reduction depends on constraint violability: two of the three constraints are violated when they conflict with a higher-ranked constraint. In this section, we show that the same sort of argument can be made for replacing OT's ranked constraints with weighted ones.

Our demonstration takes the form of a case study: ATR harmony in Lango, as described in Bavin Woock & Noonan (1979), from which all the data below are taken. Our analysis is based on generalisations originally uncovered by Bavin Woock & Noonan, and draws heavily on the analyses of Archangeli & Pulleyblank (1994) and Smolensky (2006).[6] Smolensky's use of local constraint conjunction drew our attention to the possibility of a treatment in terms of weighted constraints. In §5.2, we argue that the HG analysis improves on the earlier ones: its central principles are more general, and its typological predictions are more restrictive. Although the constraints in our analysis are simple, their interaction is complex; a correct weighting must simultaneously meet a host of conditions. Finding such a weighting involves extensive calculation. This analysis thus also further illustrates the utility of OT-Help for conducting linguistic analysis in HG.

5.1 Cumulative constraint interaction in Lango

Lango has a ten-vowel system, with five ATR vowels [i e u o ə] and five corresponding RTR vowels [ɪ ɛ ʊ ɔ a]. The following examples of ATR

[6] Other descriptions of Lango include Okello (1975) and Noonan (1992). We follow Archangeli & Pulleyblank (1994)'s characterisation of Bavin Woock & Noonan's description so as to facilitate a comparison of our analysis with previous ones. However, it is worth noting a few relevant issues in the data that should be investigated in future research. Okello (1975: 16ff) explicitly denies that right-to-left harmony is limited to high vowel triggers, provides examples of two suffixes with mid vowels that trigger harmony and claims that the failure of a mid vowel to trigger is morphologically determined. Harmony seems to be, in general, more pervasive in the dialect she describes: it is iterative and affects prefixes (cf. Bavin Woock & Noonan 1979, Noonan 1992). Both Okello and Noonan describe the blocking pattern of intervocalic consonants differently from Archangeli & Pulleyblank and Bavin Woock & Noonan, claiming that suffix-initial consonants, rather than clusters, block. Finally, both Okello and Noonan describe the harmony as strictly ATR spreading. The examples of RTR harmony cited by Archangeli & Pulleyblank occur only with a single suffix, the infinitive. Bavin Woock & Noonan also cite several examples of morphological conditioning of infinitival suffix selection with RTR roots. Since the RTR harmony data are particularly unclear, we focus only on ATR harmony.

spreading show that it targets RTR vowels in both suffixes (17a–d) and roots (17e–h), in other words, that ATR spreads left-to-right and right-to-left. We have omitted tone from all transcriptions.

(17)	a.	/wot+ɛ/	[wode]	'son (3 SG)'
	b.	/ŋut+ɛ/	[ŋute]	'neck (3 SG)'
	c.	/wot+a/	[wodə]	'son (1 SG)'
	d.	/buk+na/	[bukkə]	'book (1 SG)'
	e.	/atɪn+ni/	[atinni]	'child (2 SG)'
	f.	/dɛk+ni/	[dekki]	'stew (2 SG)'
	g.	/lʊt+wu/	[lutwu]	'stick (2 PL)'
	h.	/lɛ+wu/	[lewu]	'axe (2 PL)'

These examples also show that ATR spreads from high vowel triggers (17b, d–h) as well as from mid vowels (17a, c), and from both front (17e, f) and back vowels (17a–d, g, h). The examples also show that it crosses consonant clusters (17d–g) and singletons (17a–c, h). Finally, they show that it targets high vowels (17e, g), mid vowels (17a, b, f, h) and low vowels (17c, d).

For each of these options for trigger, directionality, intervening consonant and target, there is a preference, which is instantiated in the absence of spreading when that preference is not met. The preferences are listed in (18), along with examples of the failure to spread under dispreferred conditions, as well as references to the minimally different examples in (17) in which ATR spreading does occur in the preferred environment.

(18) *Conditions favouring ATR-spreading in Lango*

a. *High vowel trigger*

i. R-L spreading only when the trigger is high
/nɛn+Co/ [nɛnno] *[nenno] 'to see' cf. (17e–h)

ii. L-R spreading across a cluster only when the trigger is high
/gwok+na/ [gwokka] *[gwokkə] 'dog (1 SG)' cf. (17c)

b. *L-R directionality*[7]

i. Mid vowel triggers spread only L-R
/lɪm+Co/ [lɪmmo] *[limmo] 'to visit' cf. (17a, c)

ii. Spreading from a back trigger across a cluster to a non-high target only L-R
/dɛk+wu/ [dɛkwu] *[dekwu] 'stew (2 PL)' cf. (17d)

[7] The greater strength of L-R spreading also seems to be instantiated in the fact that it iterates and thus targets vowels non-adjacent to the original trigger, while R-L spreading iterates only optionally (Bavin Woock & Noonan 1979, Poser 1982, Noonan 1992, Kaplan 2008). Like Archangeli & Pulleyblank (1994) and Smolensky (2006), we abstract from the iterativity-directionality connection here, though see Jurgec (2009) for a treatment of iterativity in vowel harmony that appears compatible with our analysis.

c. *Intervening singleton*

i. L-R spreading from mid vowels occurs only across a singleton
/gwok+na/ [gwokka] *[gwokkə] 'dog (1 SG)' cf. (17a, c)

ii. R-L spreading from a back trigger to a non-high target only across a singleton
/dɛk+wu/ [dɛkwu] *[dekwu] 'stew (2 PL)' cf. (17h)

d. *High target*

R-L spreading from a back trigger across a cluster only to high vowels[8]
/dɛk+wu/ [dɛkwu] *[dekwu] 'stew (2 PL)' cf. (17g)

e. *Front trigger*

R-L spreading across a cluster to a mid target only from a front trigger
/dɛk+wu/ [dɛkwu] *[dekwu] 'stew (2 PL)' cf. (17f)

We would like an account of the harmony pattern that encodes each of these preferences with a single constraint. No such account currently exists in either OT or in rule-based approaches, as we discuss in §5.2. We now show that such an account is available under the assumption that constraints are weighted.

We follow Smolensky (2006) in ascribing the Lango trigger and directionality preferences to constraints on the heads of feature domains, though our implementation differs somewhat in the details. Headed domain structures for ATR are illustrated in (19b) and (19d), in which the ATR feature domain spans both vowels. In (19b) the head is on the rightmost vowel, and in (19d) the head is leftmost. Unlike Smolensky (2006), we assume that a feature domain is minimally binary – a relation between a head and at least one dependent. In the disharmonic sequences in (19a) and (19c), the ATR feature is linked to a single vowel, and there is no head–dependent relation. The assumption that the ATR vowels in (19a) and (19c) are not domain heads is crucial to our definition of the constraints on triggers below. In these representations, a vowel unspecified for ATR is RTR; the use of underspecification here is purely for convenience.

(19) *ATR structures*

a.	b.	c.	d.
ATR	ATR	ATR	ATR
p ɛ t i	p e t i	p e t ɪ	p e t i

[8] Noonan (1992) notes that, for some speakers, mid vowels do assimilate to following high back vowels across a cluster. This pattern can be straightforwardly accommodated by a different weighting of our constraints, for example, one just like that in Table I, but with the weights of both HEAD[front] and ATR[high] decreased to 1.

We assume that it is definitional of the head of the domain that it is faithful to its underlying specification: a head of an ATR domain is underlyingly ATR.

For spreading to occur, there must be a constraint that disprefers representations like those in (19a) and (19c) relative to (19b) and (19d) respectively. We adopt a single constraint that penalises both (19a) and (19c): Spread[ATR] (see Wilson 2003, Smolensky 2006, Jurgec 2009 and McCarthy 2009 for alternative formulations of a spreading constraint).

(20) Spread[ATR]

For any prosodic domain *x* containing a vowel specified as ATR, assign a violation mark to each vowel in *x* that is not linked to an ATR feature.

Since ATR harmony applies between roots and suffixes in Lango, the domain *x* in (20) must include them and exclude prefixes.

The transformation of an underlying representation like (19a) into a surface representation like (19b) is an instance of R-L spreading, which is dispreferred in Lango. The representation in (19b) violates the constraint in (21).[9]

(21) Head-L

Assign a violation mark to every head that is not leftmost in its domain.

For underlying (19a), Head-L and Spread[ATR] conflict: Spread [ATR] prefers spreading, as in (19b), while Head-L prefers the faithful surface representation (19a).

The transformation of an underlying representation like (19c) into a surface representation like (19d) is an instance of spreading from a mid trigger, which is also dispreferred in Lango. This violates the constraint in (22), which also conflicts with Spread[ATR].

(22) Head[high]

Assign a violation mark to every head that is not high.

[9] Baković (2000) and Hyman (2002) claim that preferences for L-R harmony are always morphologically conditioned. A more typologically responsible analysis might replace Head-L with a constraint demanding that heads be root vowels, since R-L harmony in Lango always targets root vowels. Some support for this analysis comes from the dialect of Lango described by Okello (1975), in which prefixes undergo harmony, but do not trigger it. We use Head-L for ease of comparison with Archangeli & Pulleyblank (1994) and Smolensky (2006).

Similarly, front triggers are preferred by HEAD[front].

(23) HEAD[front]
Assign a violation mark to every head that is not front.

As for the constraint preferring spreading across singleton consonants, we follow Archangeli & Pulleyblank (1994) in invoking a locality constraint.

(24) LOCAL-C
Assign a violation mark to every cluster intervening between a head and a dependent.

And finally, as the constraint penalising spreading to a non-high target, we follow Archangeli & Pulleyblank (1994) and Smolensky (2006) in using a co-occurrence constraint.

(25) ATR[high]
Assign a violation mark to every ATR vowel that is not high.

With this large set of markedness constraints that can conflict with the pro-spreading constraint SPREAD[ATR], faithfulness constraints are not necessary to characterise the patterns of blocking and spreading we have examined, and so we use only markedness constraints in the analysis we present here. A complete analysis would also include the faithfulness constraints violated by spreading (e.g. IDENT[ATR]) and faithfulness constraints that penalise alternative means of satisfying SPREAD[ATR] (e.g. MAX for segment deletion). We exclude these for reasons of space only.

Like Smolensky (2006), we consider as inputs all bisyllabic sequences containing one ATR and one RTR vowel. The potential trigger ATR vowel is either high front [i], high back [u] or mid [e]. The potential target RTR vowel is either high [ɪ] or mid [ɛ]. We illustrate the analysis with just this subset of the vowels to make the presentation as clear as possible; some of the exact combinations are not attested in (17) and (18) or in Bavin Woock & Noonan (1979) (e.g. the potential mid trigger is in fact [o] in (17) and (18)). For each ATR/RTR pair, we consider sequences with both orderings of the vowels, and for each of these, we consider inputs with intervening singletons and clusters. For each of these inputs, we consider two candidates: the faithful one, and one in which the input RTR vowel surfaces as ATR. The unfaithful candidates are assumed to have the structure illustrated in (19b, d), where the underlying RTR vowel is parsed as the dependent in the ATR domain.

In Table I, we provide a subset of the inputs, chosen for reasons we discuss below, along with the two candidates. The optimal form is labelled the winner, and the suboptimal candidate is labelled the loser (Prince 2002). A 'W' in a constraint column indicates that the constraint favours the winner, and an 'L' indicates that the constraint favours the loser. All of the constraints assign maximally one violation, so a constraint that favours the winner is violated once by the loser, and a constraint that favours the loser is violated once by the winner. The SPREAD[ATR] constraint assigns a W when the optimal form has undergone spreading, and an L when the optimal form does not. All of the other constraints assign Ls in some cases of spreading, and Ws in some cases when the candidate with spreading is suboptimal.

			11	8	4	4	2	2	
	input	W ~ L	SPREAD [ATR]	HEAD [high]	HEAD-L	LOCAL-C	HEAD [front]	ATR [high]	
T1	iCɛ	iCe ~ iCɛ	W					L	9
T2	uCɛ	uCe ~ uCɛ	W				L	L	7
T3	eCɛ	eCe ~ eCɛ	W	L				L	1
T4	ɛCi	eCi ~ ɛCi	W		L			L	5
T5	ɛCu	eCu ~ ɛCu	W		L		L	L	3
T6	ɪCe	ɪCe ~ iCe	L	W	W				1
T7	iCCɛ	iCCe ~ iCCɛ	W			L		L	5
T8	uCCɛ	uCCe ~ uCCɛ	W			L	L	L	3
T9	eCCɪ	eCCɪ ~ eCCi	L	W		W			1
T10	ɛCCi	eCCi ~ ɛCCi	W		L	L		L	1
T11	ɛCCu	ɛCCu ~ eCCu	L		W	W	W	W	1
T12	ɪCCu	iCCu ~ ɪCCu	W		L	L	L		1

Table I

Informative winner–loser pairs for Lango vowel harmony, with constraint weights and margins of separation.

There is no OT ranking of these constraints that will correctly make all of the winners optimal. None of the constraints prefers only winners, and so Recursive Constraint Demotion will immediately stall.

The topmost row shows the weights found by submitting these winner–loser pairs to the implementation of our linear programming-based solver in OT-Help. The rightmost column shows the resulting margin of separation between the optimum and its competitor, i.e. the difference between the harmony scores of the winner and the loser. Since, in this case, the constraints assign a maximum of one violation, the difference between the violation score of a winner and a loser on a given constraint is at most 1. Therefore, the margin of separation is simply the sum of the weights of the constraints that prefer the winner minus the sum of the weights that prefer the loser. The fact that these numbers are always positive shows that winners are correctly optimal under this weighting.[10]

The first six winner–loser pairs contrast L-R spreading and R-L spreading across an intervening singleton. The first three are input configurations that can yield L-R spreading, since the ATR vowel is on the left. Spreading is always optimal, even with a target mid vowel, which violates ATR[high] when it harmonises. We have left out inputs with potential target high vowels, since with this constraint set, if spreading targets a mid vowel, it is guaranteed to target high vowel in the same context. ATR[high] penalises spreading to mid vowels, and there is no constraint that specifically penalises spreading to high vowels.

The next three inputs (T4–6) are ones that can yield R-L spreading, since the ATR vowel occurs in the second syllable. Spreading in fact occurs with high triggers (T4–5), but not mid ones (T6). To illustrate the case in which spreading fails to occur, we include only an input with a potential high target /ɪ/, since, if a high vowel in a certain environment fails to undergo spreading, a mid vowel is guaranteed to fail as well.

The blocking of spreading in T6 is due to the joint effects of HEAD[high] and HEAD-L: the sum of their weights is greater than the weight of SPREAD[ATR]. An analysis in terms of such a gang effect is necessary because neither HEAD[high] alone (as in T3) nor HEAD-L alone (as in T4 and T5) is sufficient to override spreading. This is thus one source of difficulty for an OT analysis with these constraints: if either HEAD[high] or HEAD-L were placed above SPREAD[ATR] to account for T6, the wrong outcome would be produced for one of T3–5.

Inputs T7–9 provide the conditions for L-R spreading across a cluster. Spreading is blocked with a mid trigger (T9), in contrast to L-R spreading across a singleton (T3). Again, we include only the input with the potential high target to illustrate blocking, since spreading to a mid target violates a proper superset of the constraints. Blocking here is due to the combined effects of HEAD[high] and LOCAL-C, whose summed weights exceed that of SPREAD[ATR]. That LOCAL-C alone does not override SPREAD[ATR] is shown in T7–8. Again, since cumulative interaction is needed to get the correct outcome with this constraint set, OT ranking is not sufficiently powerful to deal with this set of winner-loser pairs.

Finally, inputs T10–12 illustrate the least preferred context for spreading: when the ATR vowel is on the right, and a cluster intervenes. Here, and in no other context, spreading is blocked if the trigger is back and the target is mid. This outcome is shown in T11, which can be compared with T2, 5 and 8, in which spreading does occur in other contexts. This is a gang effect between four constraints, HEAD-L, LOCAL-C, HEAD[front] and ATR[high], whose summed weight exceeds that of

[10] A display of this type is available in OT-Help as the 'comparative view'. In lieu of Ws and Ls, the HG comparative view uses positive and negative integers respectively.

SPREAD[ATR]. That no set of three of these constraints is sufficiently potent to overcome SPREAD[ATR] is illustrated by inputs T5, 8, 10 and 12, whose optimal outputs have spreading that violates one of the four possible three-membered sets of these constraints. We do not include potential mid triggers in the set of inputs, since R-L spreading already fails to occur across a singleton (T6), and spreading across a cluster also violates LOCAL-C.

In sum, the cumulative effect of any of the following three sets of constraints overcomes the demands of SPREAD[ATR].

(26) a. HEAD[high], HEAD-L
No R-L spreading from mid vowels.
b. HEAD[high], LOCAL-C
No spreading from mid vowels across a cluster.
c. HEAD-L, LOCAL-C, HEAD[front], ATR[high]
No R-L spreading from back vowels across a cluster to a mid vowel target.

No other set of constraints that does not include all of the members of one of the sets in (26) is sufficiently powerful to override SPREAD[ATR]: spreading occurs in all other contexts. A correct constraint weighting must simultaneously meet the conditions that the sum of the weights of each of the sets of constraints in (26) exceeds the weight of SPREAD[ATR], and that the sum of the weights of each of these other sets of constraints is lower than the weight of SPREAD[ATR]. OT-Help allows such a weighting to be found easily.

5.2 Comparison with alternatives

If the constraints in the previous section were considered either inviolable, as in theories outside of HG and OT, or rankable, as in OT, they would be insufficient for analysis of the Lango paradigm. In this section, we consider extant analyses constructed under each of these assumptions about the activity of constraints. We show that they suffer in terms of both generality and restrictiveness.

In their parametric rule-based analysis, Archangeli & Pulleyblank (1994) posit five rules of ATR spreading. Each rule specifies directionality and optional trigger, target and locality conditions. These are schematised in Table II. Cells left blank indicate that the rule applies with all triggers, targets or intervening consonants.

The conditions are inviolable constraints on the application of the rules. Because of their inviolability, they must be limited to apply only to particular rules: none of them are true generalisations about ATR spreading in the language as a whole. Even though the directionality, trigger and locality preferences do not state completely true generalisations, they have

direction	trigger	target	locality
L-R			VCV
L-R	high		
R-L	high		VCV
R-L	high	high	
R-L	high, front		

Table II

The rules of Archangeli & Pulleyblank (1994), each of which specifies directionality and optional trigger, target and locality conditions. Cells left blank indicate that the rule applies with all triggers, targets or intervening consonants.

broad scope in the ATR system of Lango, and must therefore be encoded as constraints on multiple rules. Thus inviolability entails the fragmentation of each generalisation across separate formal statements.

By encoding the conditions as parametric options for rules, Archangeli & Pulleyblank succeed in relating them at some level, but, in the actual statement of the conditions on spreading in Lango, there is a clear loss of generality in comparison with our weighted constraint reanalysis.[11] We can further note that there exists no proposal for how a learner sets such parameters for spreading rules (see Dresher & Kaye 1990 on metrical parameters). Correct weights for our constraints can be found not only with linear programming's simplex algorithm, but also with the Perceptron update rule (Pater 2008; see also Boersma & Pater 2008) and a host of other methods developed for neural modelling and machine learning.

Along with this loss of generality, there is a loss of restrictiveness.[12] In Archangeli & Pulleyblank's parametric rule system, any set of rules with any combination of conditions can coexist in a language. Davis (1995) and McCarthy (1997) discuss this aspect of the theory with respect to disjoint target conditions on two RTR-spreading rules; here we consider the further possibilities introduced by trigger and locality conditions. One notable aspect of the Lango system is that L-R spreading is 'stronger' in all respects: there is no environment in which R-L spreading applies more

[11] In one respect, Archangeli & Pulleyblank (1994) and Smolensky (2006) aim to generalise further than we do: to derive high vowel trigger restrictions in ATR harmony from the unmarkedness of ATR on high vowels. Pater (2009a) questions this move, pointing out that some harmony systems spread preferentially from marked vowels. John McCarthy (personal communication) notes that the strength of high triggers likely results from the greater advancement of the tongue root in high vowels. We formally encode this irreducible phonetic fact as the HEAD[high] constraint.

[12] The large space of possibilities afforded by the parametric theory is the impetus behind the development of Archangeli & Pulleyblank's own OT analysis of Lango, whose notion of 'trade-offs' may be seen as a sort of a precedent to our HG treatment.

freely with respect to any of the conditions. This 'uniform strength' property is predicted by the HG analysis, but not by the one using parametric rules. As Davis and McCarthy show, the latter theory allows one rule to apply more freely with respect to one condition, and another rule to apply more freely with respect to another condition. For example, with the following parameter settings, L-R spreading targets only high vowels, while R-L spreading has only high vowels as triggers. The set of triggers is unrestricted for L-R spreading, whereas the set of targets is unrestricted for R-L spreading.

direction	trigger	target
L-R		high
R-L	high	

Table III

Parameter setting in which L-R spreading targets only high vowels, while R-L spreading has only high vowels as triggers.

To see that this system is impossible in HG, we can consider the required weighting conditions. Along with HEAD-L, violated by R-L spreading, we include in our constraint set HEAD-R, which penalises L-R spreading. The weighting conditions are illustrated in Table IV, using the comparative format.

input	W ~ L	SPREAD[ATR]	HEAD[high]	HEAD-L	HEAD-R	ATR[high]
e…ɪ	e…i ~ e…ɪ	W	L		L	
i…ɛ	i…ɛ ~ i…e	L			W	W
ɛ…i	e…i ~ ɛ…i	W		L		L
ɪ…e	ɪ…e~ i…e	L	W	W		

Table IV

Inconsistent weighting conditions for a hypothetical pattern.

L-R spreading is illustrated in the top two rows: ATR can spread from a mid vowel, violating HEAD[high], but not to a mid vowel, which would violate ATR[high]. R-L spreading, on the other hand, can violate ATR[high], as in the third row, but not HEAD[high], as in the last one.

Recall that for the winners to be correctly optimal, in each row the sum of the weights of the constraints assigning Ws must be greater than the sum of the weights of the constraints assigning Ls. The resulting inequalities are in fact inconsistent. When this problem is submitted to OT-Help, it returns a verdict of infeasible.

By imposing other combinations of conditions on parameterised rules, there is a range of systems that one can create in which R-L spreading is stronger in one respect, and L-R is stronger in another. None of these can be generated by weightings of our constraints, since they always require inconsistent weighting conditions like those illustrated in Table IV. The general inability of HG to generate a system of this type can be understood as follows.[13] If there is a condition on spreading that applies in one direction but not another, then the sum of the weights of the constraints violated by spreading in the banned direction must be greater than the sum of the weights violated by spreading in the allowed direction (since only the former can exceed the constraint(s) motivating spreading, like our Spread). By assumption, the constraints violated under any target, trigger or locality condition are the same for both directions of spreading. Therefore, this requirement reduces to the statement that the weight of the constraint(s) violated specifically by spreading in the banned direction (e.g. Head-R) must be greater than in the permitted one (e.g. Head-L). From this it should be clear why imposing a second condition on spreading that holds only in the opposite direction would result in inconsistency amongst the weighting conditions.

Smolensky (2006)'s analysis of Lango in terms of conjoined constraints pursues a similar strategy to that of Archangeli & Pulleyblank (1994). Since OT does not allow the pattern to be analysed in terms of fully general constraints, Smolensky uses constraint conjunction to formulate complex constraints in terms of more basic formal primitives, much in the same way that Archangeli & Pulleyblank use parameterisation of rules. Again, we find the same basic constraints instantiated multiple times in the analysis, this time across conjoined constraints. To facilitate comparison with our analysis, we show this using the basic constraints from §5.1, rather than Smolensky's own.

To get spreading from high vowel triggers L-R, but not R-L, we conjoin Head[high] and Head-L. For spreading across clusters only from high vowels, we conjoin Head[high] and Local-C. Each of these conjoined constraints is violated when both of the basic constraints are violated. In Table V, we show how the conjoined constraints can resolve two of the sources of inconsistency in the failed OT analysis, using our constraint set from §5.1. In this table, the left-to-right ordering of the constraints provides a correct ranking (the dashed lines separate constraints whose ranking is indeterminate). The first two rows show the conjoined constraint analysis of spreading from mid vowels only L-R, and the second two show the analysis of spreading across clusters from only high vowels.

[13] This restriction is a generalisation of the subset criterion on targets in bidirectional spreading in OT that McCarthy (1997) attributes to personal communication from Alan Prince.

input	W ~ L	Head[high] & Head-L	Head[high] & Local-C	Spread [ATR]	Head [high]	Head-L	Local-C
eCɪ	eCi ~ eCɪ			W	L		
ɪCe	ɪCe ~ iCe	W		L	W	W	
iCCɪ	iCCi ~ iCCɪ			W			L
eCCɪ	eCCɪ ~ eCCi		W	L	W		W

Table V

The use of local conjunction to resolve inconsistency in the OT analysis of Lango.

Here Head[high] appears in three constraints, much as the high trigger condition is imposed on multiple rules in Table III. Thus the conjoined constraint analysis also succeeds only at the cost of a loss of generality relative to the weighted constraint analysis. And, like the parametric theory, there is no learning algorithm for constraint conjunction (Smolensky 2006: 139).

Furthermore, it shares with the parametric analysis the same loss of restrictiveness identified above. To show this, we provide in Table VI a local conjunction analysis of the hypothetical pattern in which only L-R spreading is triggered by mid vowels (due to conjoined Head[high] & Head-L), and only R-L spreading targets mid vowels (due to conjoined ATR[high] & Head-R).

input	W ~ L	Head[high] & Head-L	ATR[high] & Head-R	Spread [ATR]	Head [high]	Head-L	Head-R	ATR [high]
e…ɪ	e…i ~ e…ɪ			W	L		L	
i…ɛ	i…ɛ ~ i…e		W	L			W	W
ɛ…i	e…i ~ ɛ…i			W		L		L
ɪ…e	ɪ…e ~ i…e	W		L	W	W		

Table VI

The use of local conjunction to resolve inconsistency in the analysis of a hypothetical language.

For other cases in which local constraint conjunction in OT generates patterns not produced by the unconjoined versions of the basic constraints in HG, see Legendre *et al.* (2006) and Pater (to appear).

The comparison of the typological predictions of the three analyses highlights an important general point about comparisons between theories of constraint interaction, which might be easy to overlook. One might be tempted to favour a less powerful theory of constraint interaction on the grounds that it will offer a more restrictive theory of linguistic typology. However, the predictions of a theory of constraint interaction also depend on the contents of the constraint set. Insofar as a more powerful theory of constraint interaction allows attested patterns to be analysed with a more restricted constraint set, the resulting typological predictions are likely to be in some ways more restrictive. This is just as true of comparisons between HG and OT as it is of comparisons between ranked and inviolable constraints.

We offer the Lango case study as a concrete illustration of this general point. We are not asserting that it is a decisive argument in favour of HG over OT. We offer it instead in the hope that it will inspire further use of HG in linguistic analysis. There are a number of unresolved empirical issues surrounding Lango vowel harmony (see note 5) and the related typology. In recent work, McCarthy (2009) surveys the known cases in which bidirectional harmony has stronger restrictions on spreading in one direction than another, and concludes that all are doubtful for one reason or another. McCarthy's critical survey is in fact driven by the inability of his proposed constraint set to produce such patterns when they interact through OT ranking. Further cross-linguistic work driven by the current positive HG results may well yield a different outcome. Not only is further empirical study required to choose between HG and OT, but much further theoretical work is also needed to determine the ways in which HG and OT constraint sets can differ in analyses of existing languages, and the ways in which the resulting theories differ in their predictions. As we show in the following sections, OT-Help is invaluable not only in conducting analyses of individual languages in HG, but also in determining the predictions that constraint sets make in HG and OT.

6 Harmonic Grammar typology

OT provides a successful framework for the study of linguistic typology, and this has been a key component of its success. A central question is what kind of typological predictions HG makes, especially since these predictions have been claimed to be unsupported (Prince & Smolensky 1997, 2004, Legendre *et al.* 2006; cf. Pater 2009b). The present section begins to explore this question via a number of computational simulations designed to highlight points of convergence and divergence between the two frameworks. OT-Help is essential here. It allows us to explore enormous typological spaces efficiently and to compare the resulting predictions of both OT and HG.

All the data files used in these simulations are downloadable (December 2009) from http://web.linguist.umass.edu/~OTHelp/data/hg2lp/. Readers can immediately repeat our simulations using OT-Help. (A user's manual is available as Becker & Pater 2007.)

6.1 Typology calculation

In OT, a language is a set of optimal forms picked by some ranking of the constraints, and the predicted typology is the set of all the sets of optima picked by any ranking of the constraints. OTSoft (Hayes *et al.* 2003) determines the predicted typology by submitting sets of optima to the Recursive Constraint Demotion algorithm (RCDA) (Tesar & Smolensky 1998a), which either finds a ranking or indicates that none exists. OT-Help implements the RCDA as well as our linear programming approach, so we can use it to conduct typological comparisons between the two theories.

OTSoft builds up the typology by using an iterative procedure that adds a single tableau at a time to the RCDA's dataset. When a tableau is added to the dataset, the sets of optima that are sent to the RCDA are created by adding each of the new tableau's candidates to each of the sets of feasible optima that have already been found for any previously analysed tableaux. The RCDA then determines which of these new potential sets of optima are feasible under the constraint set. This procedure iterates until all of the tableaux have been added to the dataset. This is a much more efficient method of finding the feasible combinations of optima than enumerating all of the possible sets of optima and testing them all. OT-Help uses this procedure for both HG and OT.

6.2 The typology of positional restrictions

In the analysis of Lango, we pointed out that one can compare the typological predictions of HG and OT only with respect to the constraint sets that each framework requires to analyse some set of attested phenomena. In that discussion, we compared HG to OT with local constraint conjunction, showing that the less restricted constraint sets permitted by local conjunction yielded less restrictive predictions for typology. Here, we compare HG and OT using non-conjoined constraints, showing again that the greater power of HG can allow for a more restrictive theory. Our example of positional restrictions is drawn from Jesney (to appear), to which the reader is directed for a more detailed discussion; our aim here is only to show how the example illustrates this general point.

Research in OT makes use of two types of constraint to analyse what seems to be a single phenomenon: the restriction of phonological structures to particular prosodic positions. These two types of constraint – positional markedness (e.g. Itô *et al.* 1995, Zoll 1996, 1998, Walker 2001, 2005) and positional faithfulness (e.g. Casali 1996, Beckman 1997, 1998, Lombardi 1999) – capture many of the same phenomena in OT, but neither is sufficiently powerful on its own to account for the full set of attested positional restrictions. In HG, however, positional markedness constraints are able to capture a wider range of patterns, making positional faithfulness unnecessary for these cases.

Positional markedness constraints directly restrict marked structures to the 'licensing' position. Given voicing as the marked feature, for example, the constraint in (27a) disprefers any surface instance of [+voice] that appears unassociated with an onset segment, and the constraint in (27b) disprefers any surface instance of [+voice] that appears unassociated with the initial syllable.

(27) a. VoiceOnset
Assign a violation mark to every voiced obstruent that is not in onset position.

b. Voice-σ_1
Assign a violation mark to every voiced obstruent that is not in the word-initial syllable.

To illustrate the differences between HG and OT, we consider a language which allows both of the contexts identified in the constraints above – i.e. onsets and word-initial syllables – to license the marked [+voice] feature. In such a language, /badnabad/ would surface as [bad.na.bat], with devoicing only in the coda in a non-initial syllable. Table VII shows how this language can be analysed in HG with our two markedness constraints and a single non-positional faithfulness constraint.

	2	2	3	
W ~ L	VoiceOnset	Voice-σ_1	Ident[voice]	
[bad.na.bat] ~ [bad.na.pat]		L	W	1
[bad.na.bat] ~ [bat.na.bat]	L		W	1
[bad.na.bat] ~ [bad.na.bad]	W	W	L	1

Table VII
A successful HG analysis of a language in which both onsets and word-initial syllables license [+voice].

As in the Lango example, the winner and loser always differ by a maximum of one violation, so we can indicate a preference for each with 'W' and 'L', instead of indicating the degree of preference numerically. The first row compares the desired optimum to an alternative that devoices all obstruents in non-initial syllables. The loser does better on Voice-σ_1, at the expense of Ident[voice]. The second row compares the winner to a loser that devoices all codas, which improves on VoiceOnset, again at the expense of Ident[voice]. These two comparisons require each of the markedness constraints to have values lower than that of the faithfulness constraint. The last row compares the winner to the fully faithful candidate, which incurs violations of both markedness constraints. This comparison requires the sum of the weights of the markedness constraints to exceed that of the faithfulness constraint. The input /bad.na.bad/ will thus surface as [bad.na.bat], provided that the individual weights of the markedness constraints are insufficient to overcome the weight of Ident[voice], but the summed weights of the markedness constraints together are. Table VII shows a successful HG analysis. In each row, the sum of the weights of the constraints preferring the winner is greater by 1 than the sum of the weights preferring the loser.

There is no OT ranking that will make the winner correctly optimal in Table VII; no constraint assigns only Ws, and so Recursive Constraint Demotion fails. Analysing this type of pattern in OT requires positional faithfulness constraints like those defined in (28).

(28) a. IDENT[voice]-ONS
Assign a violation mark to every output segment in onset position whose input correspondent differs in voicing specification.

b. IDENT[voice]-σ_1
Assign a violation mark to every output segment in the initial syllable whose input correspondent differs in voicing specification.

The OT analysis with positional faithfulness constraints is shown in Table VIII. Here, we include general *VOICE and IDENT[voice] constraints, along with the positional faithfulness constraints defined above. The left-to-right ordering of the constraints is a correct ranking (the relative ordering of the two positional faithfulness constraints is not crucial).

W ~ L	IDENT[voice] -ONS	IDENT[voice] -σ_1	*VOICE	IDENT[voice]
[bad.na.bat] ~ [bad.na.pat]	W		L	W
[bad.na.bat] ~ [bat.na.bat]		W	L	W
[bad.na.bat] ~ [bad.na.bad]			W	L

Table VIII

A successful OT analysis using positional faithfulness to license [+voice] in both onsets and word-initial syllables.

While positional faithfulness constraints are required in OT to capture this pattern of licensing in onset and initial syllables, there are other domains where positional faithfulness constraints pose problems. A version of OT with positional faithfulness makes incorrect predictions regarding the realisation of 'floating features' and other derived structures, for example, wrongly preferring them to target weak positions (Ito & Mester 2003, Zoll 1998). To see this, we consider an input with a voice feature introduced by a second morpheme (/VCE + katnakat/). The desired optimum in this sort of case would realise the feature in a strong position where it is generally licensed – e.g. [gatnakat], with voicing surfacing on the initial onset. This is the outcome predicted by positional markedness, but not by positional faithfulness, as Table IX shows. Positional faithfulness constraints prefer that floating marked features be realised in contexts that are not normally licensers, like the non-initial coda in the loser [kat.na.kad].

W ~ L	IDENT[voice] -ONS	IDENT[voice] -σ_1	VOICE ONSET	VOICE -σ_1
[gat.na.kat] ~ [kat.na.kad]	L	L	W	W

Table IX

A situation in which positional markedness constraints are required in OT.

Cases like these, where positional faithfulness and positional markedness each account for a subset of the attested phenomena, have led to a version of OT that includes both types of constraint. A simple continuation of the examples above illustrates the typological consequences. We submitted tableaux for each of the inputs /badnabad/ and /VCE + katnakat/ to OT-Help. For HG, we included only the positional markedness constraints, along with *VOICE and IDENT[voice], while for OT we also included the positional faithfulness constraints. The results are given in Table X. The potentially optimal outputs for /badnabad/ are shown in the first column, and the potentially optimal outputs for /VCE + katnakat/ are shown in the top row. Cells are labelled with the name of the theory that makes the row and column outputs jointly optimal.

	[gat.na.kat]	[kad.na.kat]	[kat.na.gat]	[kat.na.kad]
[bad.na.bad]	HG & OT	OT	OT	OT
[bad.na.bat]	HG	OT	OT	OT
[bad.na.pat]	HG & OT	OT	OT	OT
[bat.na.bat]	HG & OT	OT	OT	OT
[bat.na.pat]	HG & OT	OT	OT	
[pat.na.pat]	HG & OT	OT	OT	OT

Table X

Typological predictions for HG with only positional markedness constraints and OT with both positional markedness and positional faithfulness constraints.

The HG results with positional markedness seem to match what is generally found typologically. The full typology may not be found for obstruent voicing, but it is found across the larger set of cases that includes positional restrictions and floating feature behaviour for other structures (see Jesney, to appear for documentation). OT with both positional faithfulness and positional markedness predicts that floating features can dock on any of the four positions defined by the two parameters initial *vs.* non-initial syllable and onset *vs.* non-onset. Thus, all of the docking sites for /VCE + katnakat/ can be made optimal, indicated in Table X by the label OT in all columns. In addition, there is practically no predicted relation between the positions in which a feature is generally permitted and where floating feature docking will occur. For example, this version of OT can generate a language in which voicing is generally restricted to onsets (/badnabad/, [bat.na.bat]), but in which a floating [+voice] feature docks onto either a final coda (/VCE + katnakat/, [kat.na.kad]) or a medial one (/VCE + katnakat/, [kad.na.kat]).

Further research is required to determine whether a version of HG without positional faithfulness constraints can indeed deal with the full range of phenomena attributed to these constraints in OT. These initial results suggest that the pursuit of such a theory may yield a resolution to a long-standing problem in OT. Furthermore, since there is not a subset relation in the types of languages generated by the two theories of constraints and constraint interaction illustrated in Table X, this example illustrates the general point that a fleshed-out theory of some set of attested phenomena in HG will likely be in some ways both less restrictive and more restrictive than an OT one.

6.3 Gradient Alignment and Lapse constraints

We now turn to an example concerning the typological spaces determined by two different classes of constraint that have been used for stress typology in OT. McCarthy & Prince (1993) propose an account of stress placement in terms of Alignment constraints, which demand coincidence of edges of prosodic categories. Gradient Alignment constraints are ones whose degree of violation depends on the distance between the category edges: roughly, if x should be at, say, the leftmost edge of a certain domain and it surfaces n segments (syllables) from that edge, then x incurs n violations for the candidate containing it. Kager (2005) proposes an alternative account of stress placement in OT that replaces gradient Alignment constraints with a set of Lapse constraints, which penalise adjacent unstressed syllables in various environments, assigning one mark per violation, as with normal markedness constraints.

To examine the typological predictions of the two accounts, Kager constructed OTSoft files (Hayes *et al.* 2003) with a set of candidate parsings for words from two to nine syllables in length. Separate files contained the appropriate violation marks for each constraint set. For each of these, there were separate files for trochaic (left-headed) feet and for iambic (right-headed) feet (here we discuss only the trochaic results). Using OTSoft, Kager found that the gradient Alignment constraint set generated 35 languages, while the one with Lapse constraints generated 25.

We used OT-Help to replicate Kager's experiment, using both OT and HG. The results for the two constraint sets discussed above, derived from OTSoft files prepared by Kager, are shown in Table XI. We provide the number of languages that each combination of constraints and mode of interaction predicts, out of a total of 685,292,000 possible combinations of optima.

	OT	HG
Gradient alignment	35	911
Lapse	25	85

Table XI
Number of predicted languages.

For both constraint sets, HG generates all the languages that OT does. HG also generates a significant number of languages that OT does not.

A primary source of this dramatic increase is the manner in which gradient Alignment constraints assign violation marks. To illustrate, we show four potential parses of a six-syllable word, and the violations they incur on two constraints. Foot edges are indicated by parentheses, and prosodic word edges by square brackets. ALIGN-L(Ft, Wd) demands that the left edge of every foot be aligned with the left edge of the word and is violated by each syllable intervening between these two edges. PARSE-σ is violated by every syllable that fails to be parsed into a foot.

(29)

	ALIGN-L(Ft,Wd)	PARSE-σ
a. [(ta.ta)(ta.ta)(ta.ta)]	2+4=6	0
b. [(ta.ta)(ta.ta)ta.ta]	2	2
c. [(ta.ta)ta.ta.ta.ta]	0	4
d. [ta.ta.ta.ta.ta.ta]	0	6

ALIGN-L(Ft, Wd) and PARSE-σ conflict in that every foot added after the leftmost one satisfies PARSE-σ at the cost of violating ALIGN-L(Ft, Wd). This cost increases as feet are added: the second foot from the left adds two violations, the third one adds four and so on. This increasing cost interacts with weighting to produce a rich typology. With an appropriate weighting (e.g. a weight of 1 for ALIGN-L(Ft, Wd) and a weight of 2 for PARSE-σ), a second foot will be added to avoid violating PARSE-σ, but not a third one: (29b) emerges as optimal. This outcome would be impossible in HG, as it is in OT, if each non-leftmost foot added the same number of violations of ALIGN-L(Ft, Wd) (or whatever constraint replaces it).[14]

The HG typology with Lapse constraints is much closer to that of OT, but it still yields more than a threefold increase in predicted languages. We believe that it would be a mistake to take this sort of result to argue definitively for OT. First, it was arrived at using a constraint set designed for OT. As we have shown in §§5 and 6.2, weighted interaction allows for

[14] See McCarthy (2003) for extensive arguments for the replacement of gradient Alignment in OT.

different constraints than those used in OT, and these possibilities must be further explored to better understand the theory and how it differs from OT. Second, the result also depends on a particular mode of evaluation: here, the entire representation is evaluated once and only once by the entire set of constraints. As Pater (2009b, to appear) shows, changing assumptions about mode of evaluation yields positive results for HG typology, in addition to those that McCarthy (2006, 2007, 2009) demonstrates for OT (see also Pruitt 2008 on stress in Serial OT).

6.4 A typological correspondence between OT and HG

The previous simulation highlights the fact that OT and HG can produce quite different typological predictions. However, as we emphasised in the introduction, the two frameworks do not invariably diverge. The present section describes a simulation involving a fairly complex set of constraints for which OT and HG deliver identical typological predictions. The result is especially striking in light of the fact that some of the constraints are gradient Alignment constraints of the sort that produced a large difference in the previous section.

The simulation involves the following set of constraints.

(30) a. Trochee
Assign a violation to every right-headed foot.
b. Iamb
Assign a violation to every left-headed foot.
c. Align(Ft)-L
For every foot, assign a violation for every syllable separating it from the left edge of the word.
d. Align(Ft)-R
For every foot, assign a violation for every syllable separating it from the right edge of the word.
e. Align(Hd)-L
Assign a violation for every syllable separating the main stressed syllable from the left edge of the word.
f. Align(Hd)-R
Assign a violation for every syllable separating the main stressed syllable from the right edge of the word.

The candidate set for the simulation consisted of all logically possible parses of words of two to five syllables in length into left- and right-headed bisyllabic feet, with main stress on either one of the feet in the four- and five-syllable words. The parses are all exhaustive, up to the limits imposed by the binary minimum; there is no more than one unparsed syllable per word.

Here is a summary of the results of this simulation.

(31) *Number of predicted languages with the constraint set in* (30)

a. All logically possible combinations of optima: 1536
b. OT: 18
c. HG: 18

Not only are the counts the same, but the languages themselves are the same (OT-Help does these calculations and comparisons automatically).

An interesting aspect of this result is that the constraint set contains the gradient Alignment constraints ALIGN(Ft) and ALIGN(Hd), which, as we saw in §6.3, can lead to significant differences in the predictions of OT and HG. Crucially, however, the constraint set contains neither PARSE-σ nor WEIGHT-TO-STRESS. Because it lacks PARSE-σ, the trade-off in violations between it and ALIGN(Ft) illustrated in (29) does not exist in the current set of violation profiles. Because it lacks WEIGHT-TO-STRESS, a trade-off with ALIGN(Hd), discussed by Legendre *et al.* (2006) and Pater (2009b), is also absent. We do not take this as evidence for the elimination of WEIGHT-TO-STRESS and PARSE-σ from metrical theory. Rather, it serves to further illustrate the crucial point that it is the trade-offs between violations of constraints, rather than the way that any one constraint assigns violations, that lead to differences between HG and OT. Like the NOCODA/MAX example in the introduction, this is because the version of HG we are considering is an optimisation system.

6.5 Summary

The typological investigations above, which mix qualitative analysis of specific cases with large-scale quantitative assessment, point up the complexity of the relationship between OT and HG. There are constraint sets for which the two frameworks are aligned in their typological predictions, and there are constraint sets for which they diverge wildly. The examples show that certain constraint combinations can have apparent ill-effects in one framework even as they produce desirable patterns in the other. These findings are just small pieces in the larger puzzle of how the two approaches relate to one another. We think the connection with linear programming, and the computational tools that go with it, can facilitate rapid progress in putting the rest of the pieces together.

7 Conclusion

We have shown that Harmonic Grammar learning problems translate into linear systems that are solvable using linear programming algorithms. This is an important mathematical connection, and it has a practical component as well: our software package OT-Help facilitates comparison between weighting and other constraint-based approaches. This implementation, freely available and requiring no specialised user expertise, gets us over the intrinsic practical obstacles to exploring weighting systems. We can then focus attention on the linguistic usefulness of HG and related approaches, as we have done with our in-depth analysis of Lango ATR harmony (§5) and our typological investigations (§6).

The formal results of this paper are best summarised by drawing an explicit connection with the fundamental theorem of linear programming (Cormen *et al.* 2001: 816).

(32) **Theorem 1** (the fundamental theorem of linear programming)
If L is a linear system, then there are just three possibilities:
a. L has an optimal solution with a finite objective function.
b. L is unbounded (in which case we can return a solution, though the notion of optimal is undefined).
c. L is infeasible (no solution satisfies all its conditions).

Our method applies this theorem to understanding HG. The UNBOUNDED outcome is not directly relevant; we always solve minimisation problems, and our systems are structured so that there is always a well-defined minimum. The INFEASIBLE verdict is essential. It tells us that the current grammar cannot deliver the set of optimal candidates we have specified. This might be a signal that the analysis must change, or it might prove that a predicted typological gap in fact exists for the current constraint set. And if we are presented with an optimal solution, then we know our grammar delivers the specified set of forms as optimal. Moreover, we can then analyse the solution to learn about the relations among our constraints.

We obtain these results efficiently; though the worst-case running time for the simplex algorithm is exponential, it is extremely efficient in practice, often besting its theoretically more efficient competitors (Chvátal 1983: 4, Cormen *et al.* 2001: 820–821). What's more, we have opened the way to applying new algorithms to the problem, with an eye towards achieving an optimal fit between the structure of linguistic systems and the nature of the computational analysis. Our approach works for the full range of harmonic grammars as we define them in §2, including very large and complex ones. We therefore see the translation of HG systems into linear systems solvable using linear programming methods as providing a valuable tool for the serious exploration of constraint weighting in linguistics. We also see great promise in the approach for developing theories of learning, for determining the nature of the constraint set and for gaining a deeper mathematical and algorithmic understanding of the theory's main building blocks.

REFERENCES

Albright, Adam, Giorgio Magri & Jennifer Michaels (2008). Modeling doubly marked lags with a split additive model. In Harvey Chan, Heather Jacob & Enkeleida Kapia (eds.) *Proceedings of the 32nd Annual Boston University Conference on Language Development*. Somerville: Cascadilla. 36–47.

Archangeli, Diana & Douglas Pulleyblank (1994). *Grounded phonology*. Cambridge, Mass.: MIT Press.

Baković, Eric (2000). *Harmony, dominance, and control*. PhD dissertation, Rutgers University.

Bavin Woock, Edith & Michael Noonan (1979). Vowel harmony in Lango. *CLS* **15**. 20–29.

Bazaraa, Mokhtar S., John J. Jarvis & Hanif D. Sherali (2005). *Linear programming and network flows*. 3rd edn. Hoboken, NJ: Wiley.

Becker, Michael & Joe Pater (2007). OT-Help user guide. *University of Massachusetts Occasional Papers in Linguistics* **36**. 1–12.

Becker, Michael, Joe Pater & Christopher Potts (2007). OT-Help 1.2. Software available at http://web.linguist.umass.edu/~OTHelp/.

Beckman, Jill N. (1997). Positional faithfulness, positional neutralisation and Shona vowel harmony. *Phonology* **14**. 1–46.

Beckman, Jill N. (1998). *Positional faithfulness*. PhD dissertation, University of Massachusetts, Amherst.

Boersma, Paul & Joe Pater (2008). Convergence properties of a gradual learning algorithm for Harmonic Grammar. Available as ROA-970 from the Rutgers Optimality Archive.

Boersma, Paul & David Weenink (2009). *Praat: doing phonetics by computer* (version 5.1.12). http://www.praat.org/.

Casali, Roderic F. (1996). *Resolving hiatus*. PhD dissertation, University of California, Los Angeles.

Chvátal, Vašek (1983). *Linear programming*. New York: Freeman.

Coetzee, Andries & Joe Pater (in press). The place of variation in phonological theory. In John A. Goldsmith, Jason Riggle & Alan Yu (eds.) *The handbook of phonological theory*. 2nd edn. Oxford: Blackwell.

Cormen, Thomas H., Charles E. Leiserson, Ronald L. Rivest & Clifford Stein (2001). *Introduction to algorithms*. 2nd edn. Cambridge, Mass.: MIT Press.

Dantzig, George B. (1982). Reminiscences about the origins of linear programming. *Operations Research Letters* **1**. 43–48.

Davis, Stuart (1995). Emphasis spread in Arabic and Grounded Phonology. *LI* **26**. 465–498.

Dresher, B. Elan & Jonathan D. Kaye (1990). A computational learning model for metrical phonology. *Cognition* **34**. 137–195.

Goldsmith, John A. (1990). *Autosegmental and metrical phonology*. Oxford & Cambridge, Mass.: Blackwell.

Goldsmith, John A. (1991). Phonology as an intelligent system. In Donna Jo Napoli & Judy Anne Kegl (eds.) *Bridges between psychology and linguistics: a Swarthmore Festschrift for Lila Gleitman*. Hillsdale: Erlbaum. 247–268.

Goldsmith, John A. (1993). Introduction. In John A. Goldsmith (ed.) *The last phonological rule: reflections on constraints and derivations*. Chicago: University of Chicago Press. 1–20.

Goldsmith, John A. (1999). Introduction. In John A. Goldsmith (ed.) *Phonological theory: the essential readings*. Malden, Mass. & Oxford: Blackwell. 1–16.

Goldwater, Sharon & Mark Johnson (2003). Learning OT constraint rankings using a Maximum Entropy model. In Jennifer Spenador, Anders Eriksson & Östen Dahl (eds.) *Proceedings of the Stockholm Workshop on Variation within Optimality Theory*. Stockholm: Stockholm University. 111–120.

Hayes, Bruce, Bruce Tesar & Kie Zuraw (2003). *OTSoft 2.1*. http://www.linguistics.ucla.edu/people/hayes/otsoft/.

Hayes, Bruce, Kie Zuraw, Péter Siptár & Zsuzsa Cziráky Londe (2008). Natural and unnatural constraints in Hungarian vowel harmony. Ms, University of California, Los Angeles.

Hyman, Larry M. (2002). Is there a right-to-left bias in vowel harmony? Ms, University of California, Berkeley.

Ito, Junko & Armin Mester (2003). *Japanese morphophonemics: markedness and word structure*. Cambridge, Mass.: MIT Press.

Itô, Junko, Armin Mester & Jaye Padgett (1995). Licensing and underspecification in Optimality Theory. *LI* **26**. 571–613.

Jäger, Gerhard (2007). Maximum entropy models and Stochastic Optimality Theory. In Annie Zaenen, Jane Simpson, Tracy Holloway King, Jane Grimshaw, Joan Maling & Chris Manning (eds.) *Architectures, rules, and preferences: variations on themes by Joan W. Bresnan*. Stanford: CSLI. 467–479.

Jesney, Karen (to appear). Licensing in multiple contexts: an argument for Harmonic Grammar. *CLS* **45**.

Johnson, Mark (2002). Optimality-theoretic Lexical Functional Grammar. In Paola Merlo & Suzanne Stevenson (eds.) *The lexical basis of sentence processing: formal, computational and experimental issues*. Amsterdam & Philadelphia: Benjamins. 59–73.

Jurgec, Peter (2009). Autosegmental spreading is a binary relation. Ms, University of Tromsø.

Kager, René (2005). Rhythmic licensing: an extended typology. In *Proceedings of the 3rd International Conference on Phonology*. Seoul: The Phonology–Morphology Circle of Korea. 5–31.

Kaplan, Aaron F. (2008). *Noniterativity is an emergent property of grammar*. PhD dissertation, University of California, Santa Cruz.

Kelkar, Ashok R. (1968). *Studies in Hindi-Urdu*. Vol. 1: *Introduction and word phonology*. Poona: Deccan College.

Keller, Frank (2000). *Gradience in grammar: experimental and computational aspects of degrees of grammaticality*. PhD dissertation, University of Edinburgh.

Keller, Frank (2006). Linear optimality theory as a model of gradience in grammar. In Gisbert Fanselow, Caroline Féry, Ralf Vogel & Matthias Schlesewsky (eds.) *Gradience in grammar: generative perspectives*. Oxford: Oxford University Press. 270–287.

Legendre, Géraldine, Yoshiro Miyata & Paul Smolensky (1990a). Harmonic Grammar: a formal multi-level connectionist theory of linguistic well-formedness: theoretical foundations. In *Proceedings of the 12th Annual Conference of the Cognitive Science Society*. Hillsdale: Erlbaum. 388–395.

Legendre, Géraldine, Yoshiro Miyata & Paul Smolensky (1990b). Harmonic Grammar: a formal multi-level connectionist theory of linguistic well-formedness: an application. In *Proceedings of the 12th Annual Conference of the Cognitive Science Society*. Hillsdale: Erlbaum. 884–891.

Legendre, Géraldine, Antonella Sorace & Paul Smolensky (2006). The Optimality Theory–Harmonic Grammar connection. In Smolensky & Legendre (2006: vol. 2). 339–402.

Lombardi, Linda (1999). Positional faithfulness and voicing assimilation in Optimality Theory. *NLLT* **17**. 267–302.

López, Marco & Georg Still (2007). Semi-infinite programming. *European Journal of Operations Research* **180**. 491–518.

McCarthy, John J. (1997). Process-specific constraints in optimality theory. *LI* **28**. 231–251.

McCarthy, John J. (2003). OT constraints are categorical. *Phonology* **20**. 75–138.

McCarthy, John J. (2006). Restraint of analysis. In Eric Baković, Junko Ito & John J. McCarthy (eds.) *Wondering at the natural fecundity of things: essays in honor of Alan Prince*. Santa Cruz: Linguistics Research Center. 195–219.

McCarthy, John J. (2007). *Hidden generalizations: phonological opacity in Optimality Theory*. London: Equinox.

McCarthy, John J. (2009). Harmony in Harmonic Serialism. Ms, University of Massachusetts, Amherst. Available as ROA-1009 from the Rutgers Optimality Archive.

McCarthy, John J. & Alan Prince (1993). Generalized alignment. *Yearbook of Morphology 1993*. 79–153.

Noonan, Michael (1992). *A grammar of Lango*. Berlin & New York: Mouton de Gruyter.

Okello, Jenny (1975). *Some phonological and morphological processes in Lango*. PhD dissertation, Indiana University.

Pater, Joe (2008). Gradual learning and convergence. *LI* **39**. 334–345.

Pater, Joe (2009a). Review of Smolensky & Legendre (2006). *Phonology* **26**. 217–226.

Pater, Joe (2009b). Weighted constraints in generative linguistics. *Cognitive Science* **33**. 999–1035.

Pater, Joe (to appear). Serial Harmonic Grammar and Berber syllabification. In Toni Borowsky, Shigeto Kawahara, Takahito Shinya & Mariko Sugahara (eds.) *Prosody matters: essays in honor of Elisabeth O. Selkirk*. London: Equinox.

Poser, William J. (1982). Phonological representation and action-at-a-distance. In Harry van der Hulst & Norval Smith (eds.) *The structure of phonological representations*. Part 2. Dordrecht: Foris. 121–158.

Potts, Christopher, Michael Becker, Rajesh Bhatt & Joe Pater (2007). HaLP: Harmonic Grammar with linear programming. Version 2. Software available at http://web.linguist.umass.edu/~halp/.

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

Prince, Alan (2003). Anything goes. In Takeru Honma, Masao Okazaki, Toshiyuki Tabata & Shin-ichi Tanaka (eds.) *A new century of phonology and phonological theory: a Festschrift for Professor Shosuke Haraguchi on the occasion of his sixtieth birthday*. Tokyo: Kaitakusha. 66–90.

Prince, Alan & Paul Smolensky (1997). Optimality: from neural networks to Universal Grammar. *Science* **275**. 1604–1610.

Prince, Alan & Paul Smolensky (2004). *Optimality Theory: constraint interaction in generative grammar*. Malden, Mass. & Oxford: Blackwell.

Pruitt, Kathryn (2008). Iterative foot optimization and locality in rhythmic word stress. Ms, University of Massachusetts, Amherst.

Riggle, Jason (2004a). *Generation, recognition, and learning in finite-state Optimality Theory*. PhD dissertation, University of California, Los Angeles.

Riggle, Jason (2004b). Generation, recognition and ranking with compiled OT grammars. Paper presented at the 78th Annual Meeting of the Linguistic Society of America, Boston.

Smolensky, Paul (2006). Optimality in phonology II: harmonic completeness, local constraint conjunction, and feature domain markedness. In Smolensky & Legendre (2006: vol. 2). 27–160.

Smolensky, Paul & Géraldine Legendre (eds.) (2006). *The harmonic mind: from neural computation to optimality-theoretic grammar*. 2 vols. Cambridge, Mass.: MIT Press.

Tesar, Bruce & Paul Smolensky (1998a). Learnability in Optimality Theory. *LI* **29**. 229–268.

Tesar, Bruce & Paul Smolensky (1998b). Learning Optimality-Theoretic grammars. *Lingua* **106**. 161–196.

Walker, Rachel (2001). Positional markedness in vowel harmony. In Caroline Féry, Antony Dubach Green & Ruben van de Vijver (eds.) *Proceedings of HILP 5*. Potsdam: University of Potsdam. 212–232.

Walker, Rachel (2005). Weak triggers in vowel harmony. *NLLT* **23**. 917–989.

Wilson, Colin (2003). Analyzing unbounded spreading with constraints: marks, targets, and derivations. Ms, University of California, Los Angeles.

Wilson, Colin (2006). Learning phonology with substantive bias: an experimental and computational study of velar palatalization. *Cognitive Science* **30**. 945–982.
Zoll, Cheryl (1996). *Parsing below the segment in a constraint-based framework*. PhD dissertation, University of California, Berkeley.
Zoll, Cheryl (1998). Positional asymmetries and licensing. Ms, MIT. Available as ROA-282 from the Rutgers Optimality Archive.