

Class 5: Probability distributions over OT rankings II

To do for tomorrow (Tuesday)

- Put your own data (real or fake) in OTSoft input-file form, and bring it to the lab tomorrow.
- You may want get a head start on the upcoming required readings, Coetzee 2009 (for Thursday) and section 4.7 of Martin 2007 (for Friday).

Overview: Less-restrictive probability distributions: Stochastic OT and its learning algorithms

1 What probability distributions should a theory allow?

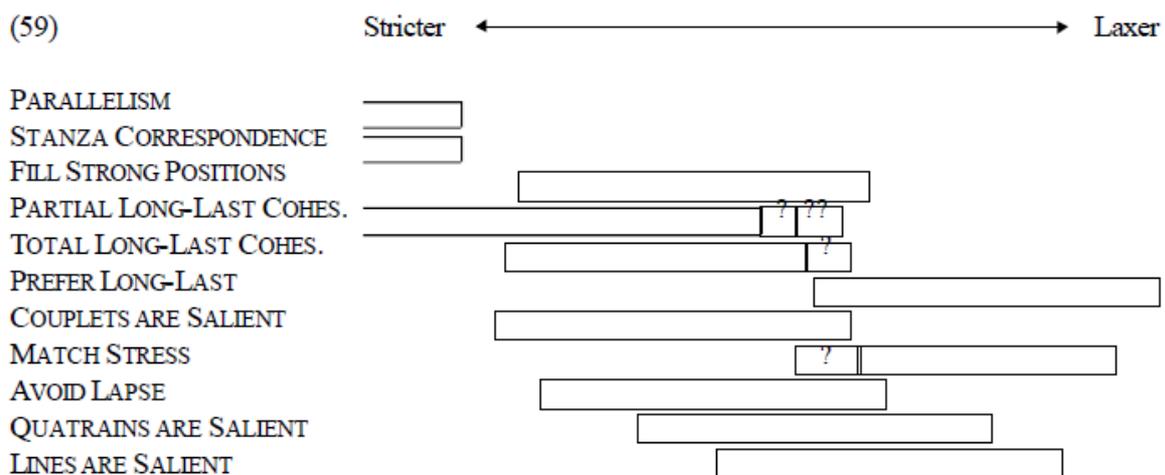
Suppose three constraints: A, B, C

6 linear orders: A>>B>>C, A>>C>>B, B>>A>>C, B>>C>>A, C>>A>>B, C>>B>>A

- Last time, we considered a totally unrestricted theory. We can attach any probabilities to the 6 rankings, as long as they add up to 1
 - 100% A>>B>>C, 0% for the rest
 - 95% A>>B>>C, 1% each for the rest
 - 50% A>>B>>C, 50% C>>B>>A, 0% for the rest
 - ...and infinitely many more
- We then saw Anttila's partial ordering model, which allows only certain distributions
 - 100% A>>B>>C, 0% for the rest (and similarly for the other 5 rankings)
 - 50% A>>B>>C, 50% B>>A>>C
 - 50% C>>A>>B, 50% C>>B>>A (and similarly for the other 2 pairs of constraints)
 - 16.7% for each ranking
- Today we look at a model that's a bit less restrictive than Anttila's, Stochastic OT
 - We won't try to decide which model is a better empirical match to the world's languages.

2 The basic idea

- Assign each constraint to a range on the number line.
 - Early version of the idea from Hayes & MacEachern. Each constraint is associated with a range, and those ranges also have fringes, indicated by "?" or "??"



(Hayes & MacEachern 1998, p. 43)

- Each time you want to generate an output, choose one point from each constraint's range, then use a total ranking according to those points.

3 Stochastic OT

Boersma 1997; Boersma & Hayes 2001

- This was the first theory to quantify ranking preference.
- “stochastic” just means “probabilistic”, so various theories could be described as “stochastic OT”. With a capital S, though, I mean specifically Boersma’s theory

As you read, in the grammar, each constraint has a “ranking value”:

*θ 101
 IDENT(cont) 99

Every time a person speaks, they add a little noise to each of these numbers, then rank the constraints according to these numbers.

⇒ Go to demo (I’ve prepared an Excel file so we can see how this works)

- Researchers who use this model often acknowledge stylistic conditioning, but idealize away from it. Ideas on how we could modify the model to add in the effect of style?

4 Stochastic OT

(Boersma 1997; Boersma & Hayes 2001)

- This was a groundbreaking aspect of the proposal: it came with a procedure for learning the values.
 - Important theoretically: if this is a theory of what a person’s grammar looks like, we need some theory of how the grammar gets that way, during childhood and beyond
 - Important practically: it meant you could apply these models to your own data

Procedure:

1. Suppose you’re a child. You start out with both constraints’ ranking values at 100.
2. You hear an adult say something—suppose /θɪk/ → [tɪk]
3. You use your current ranking values to produce an output. Suppose it’s /θɪk/ → [θɪk].
4. Your grammar produced the wrong result! (If the result was right, repeat from Step 2)
5. Constraints that [tɪk] violates are ranked too low; constraints that [θɪk] violates are too high.
6. So, promote and demote them, by some fixed amount (say 0.33 points)

	/θɪk/	*θ	IDENT(cont)
the adult said this	[θɪk]	*	
your grammar produced this	[tɪk]	demote to 99.67	* promote to 100.33

7. Repeat.

⇒ Go to demo (same Excel file, different worksheet)

- Suppose, as in our demo, that adults produce [tɪk] 90% of the time. Will your grammar ever stop making errors?
- What’s the effect of the column labeled ‘plasticity’?
- What if the adults actually don’t vary, and the outcome is always [θɪk]. What will happen to the ranking values? (After discussing, let’s try it in the spreadsheet.)
- In that case, will your grammar ever stop making errors?

5 Using the Gradual Learning Algorithm

- Fortunately, you don't need to make an Excel file like this.
- Bruce Hayes's OTSoft (Hayes & al. 2003) will do the work for you!
- You can also use Praat (Boersma & Weenink 2012)

⇒ Go to OTSoft demo

6 Some interesting options in OTSoft's GLA function

By default, all constraints start at 100. You can change that.

By default, plasticity gradually changes from the initial to the final value. You can change that.

You can specify that certain constraints must outrank others—they will be kept 20 units apart (or some other value that you specify)

The screenshot shows the OTSoft 2.3.1 software interface with the following options and callouts:

- Number of times to go through forms:** 10000. Callout: "If 0, initial rankings values never change."
- Initial plasticity:** 1. Callout: "How much does each ranking value change when there's an error (at the beginning of learning)"
- Final plasticity:** .01. Callout: "Plasticity gradually changes from initial value to final value. It's common for 'final' to be lower. The idea is that your grammar changes less as you get older/more experienced"
- Number of times to test grammar:** 100000. Callout: "After learning is finished, for each input the software will generate an output using the grammar that it learned. If it does this 100,000 times, you get a good estimate of each candidate's probability of winning under that grammar"
- Using customized initial ranking values from file:** (checkbox)
- Buttons:** "Exit to main screen" and "Run GLA"

Some useful tricks:

- To just see what probabilities are assigned to candidates under a certain grammar:
 - create a file with customized initial ranking values (use "Initial rankings" menu)
 - set "Number of times to go through forms" to 0
- To train a grammar on certain data (training data) and test it on other data (testing data):
 - In your OTSoft input file, for the testing data, give frequency of 0 to all candidates.
 - Thus they won't contribute to learning, but they'll still be used in testing

Now let's turn to some case studies

7 Albright & Hayes 2006: "Junk" constraints

Albright & Hayes 2006 is one of a series of papers developing a model for learning constraints from morphological mappings.

Navajo sibilant harmony:

(3) a.	[bà:ʔ]	[sì-bà:ʔ]	
b.	[č'ih]	[ši-č'ih]	
c.	[č ^h ò:jìn]	[ši-č ^h ò:jìn]	
d.	[gàn]	[sì-gàn]	
e.	[k'áz]	[sì-k'áz]	
f.	[kéšgã:]	[ši-kéšgã:], [sì-kéšgã:]	
g.	[sí:ʔ]	[sì-sí:ʔ]	
h.	[tãš]	[ši-tãš], [sì-tãš]	
i.	[tí]	[sì-tí]	
j.	[t̄lé:ž]	[ši-t̄lé:ž], [sì-t̄lé:ž]	(Albright & Hayes 2006, p. 3)

The Albright/Hayes learner (which we won't get into) learns some sensible constraints like these:

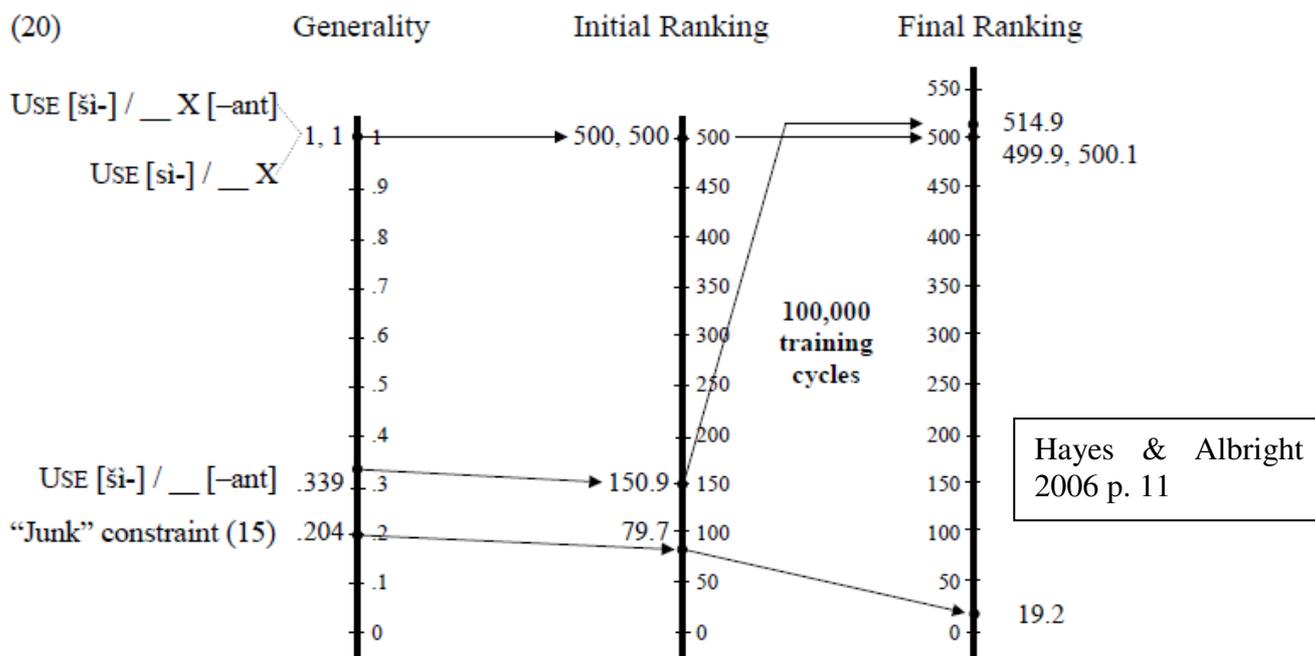
- USE ŠI / __[-anterior] local harmony: should be high-ranked
- USE ŠI / __X[-anterior] distal harmony: should be mid-ranked
- USE SI default is [+ant] should be lower-ranked

but also some "junk" constraints like these:

- USE SI / __([-round])* [+ant, +cont] ([-cons])* # happens to be true in training data but probably not high-ranked in real grammar

==> Demo: let's see what happens if we apply GLA to a schematic case like this

- Albright & Hayes's solution:
 - constraints' **initial ranking values** reflect their **generality**
 - generality of USE ŠI / __[-anterior] = 19 (# of __[-anterior] words) / 56 (# of words that use ŠI) = 0.34
 - generality of junk constraint = 37/181 = 0.20
 - These numbers are then scaled so that they range from 0 to 500 (see paper for details):



==> Let's try this in our demo

- Why does it work?
 - USE SI is ranked high enough from the beginning to avoid errors like /si+tala/ → [šitala]
 - So, the junk constraint never gets promoted
- Albright and Hayes would probably both favor a MaxEnt approach now (Friday's class),
 - but this is a nice demonstration of how to introduce bias into a learner
 - In this case, the desired bias is not to put too much trust into constraints that cover only few cases

8 Boersma & Level 2000: predicting acquisition order

- The G in GLA stands for "gradual"
 - The algorithm doesn't just return its final grammar
 - Instead, it gradually updates the initial grammar
 - At every step, it's possible to "pause" the grammar and ask what its current output is
 - This provides a concrete analogy to child language acquisition
- Levelt had previously done work on the order in which different syllable types are produced by children.
 - Here are data for 12 children acquiring Dutch:

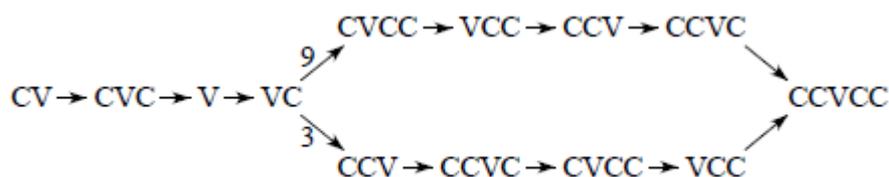


Figure 1. Acquisition order for syllable types in Dutch.

(Boersma & Levelt, p.1)

- Can we get the GLA to replicate this?
- Boersma & Levelt fed the GLA the frequencies of the faithful syllables in Dutch
 - They also, importantly, set markedness constraints' initial ranking value to 100, and faithfulness constraints' to 50

- This means that initially, all outputs will be [CV], regardless of input
- Gradually, some faithfulness constraints will climb (they just use one, FAITH), some markedness constraints will fall, and some other syllable shapes will get produced

==> DEMO—let's try this with different amounts of learning

Boersma & Levelt's resulting ranking values over time

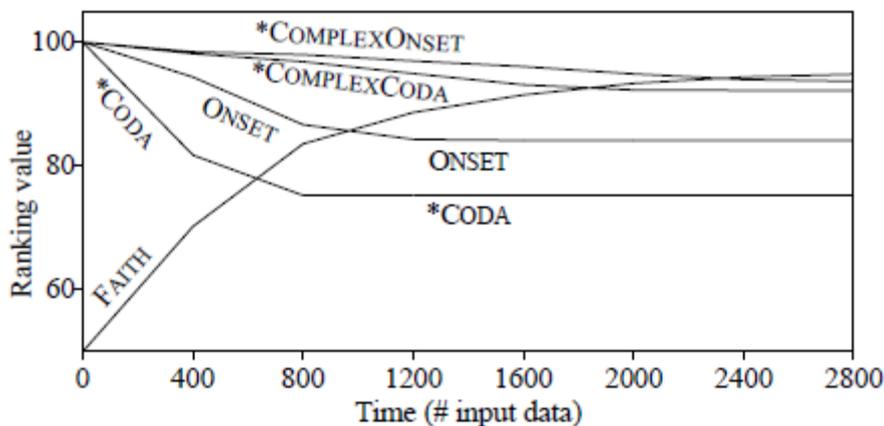


Figure 2. Constraint rankings as functions of time.

(Boersma & Levelt p. 5)

Here are the rates of correct (faithful) production for each syllable type over time

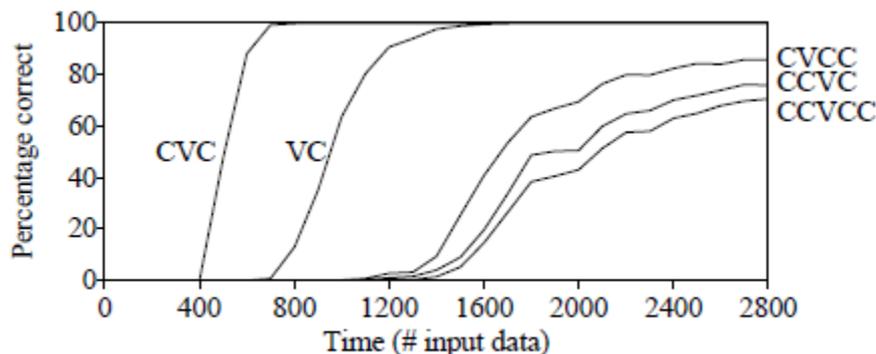


Figure 3. Five learning curves for our simulated learner.

(Boersma & Levelt p. 7)

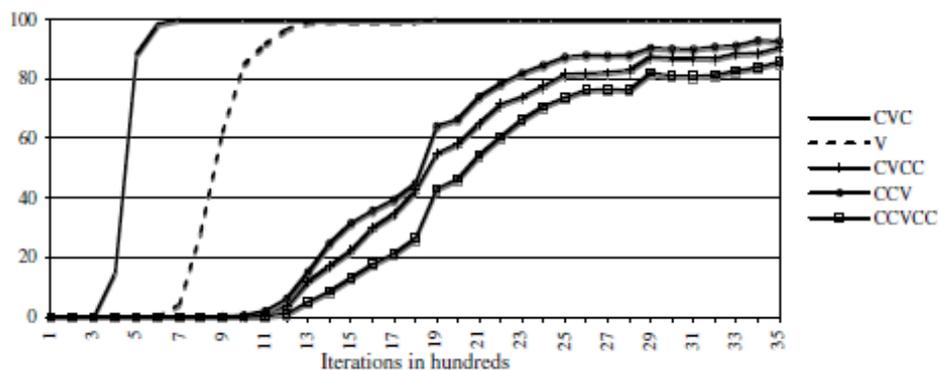
- Jarosz 2010 shows how the frequencies of the different syllable types in the language matter
 - if a syllable type is rare, then errors on it are always rare
 - ...even if the accuracy of the current grammar on these syllable types is low
 - A markedness constraint's demotion rate depends on how many word tokens violate it
- Here are Jarosz's results (following the Boersma & Levelt procedure) for Dutch, English, and Polish
 - Same constraints, faithful candidate is always the winner—but the input frequencies differ.

(results on next page)

TABLE 1. *Relative frequencies of syllable types in Dutch*

CV	CVC	CVCC	V	VC	VCC	CCV	CCVC	CCVCC
44.8%	32.1%	3.3%	3.9%	12.0%	0.4%	1.4%	2.0%	0.3%

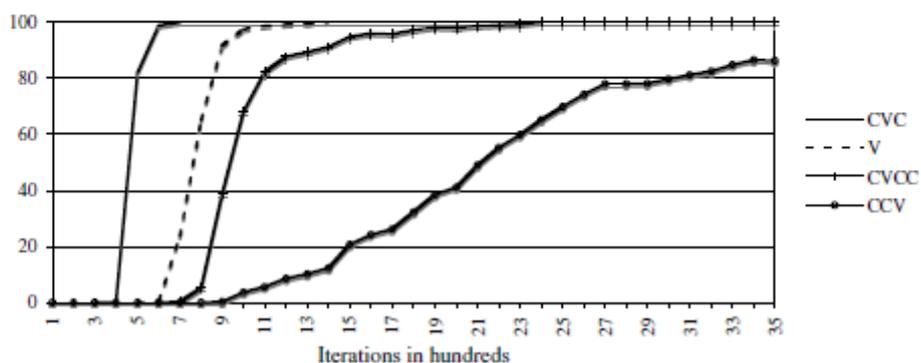
Jarosz p. 573



Dutch, Jarosz p. 594

TABLE 5. *Relative frequencies of syllable types in English*

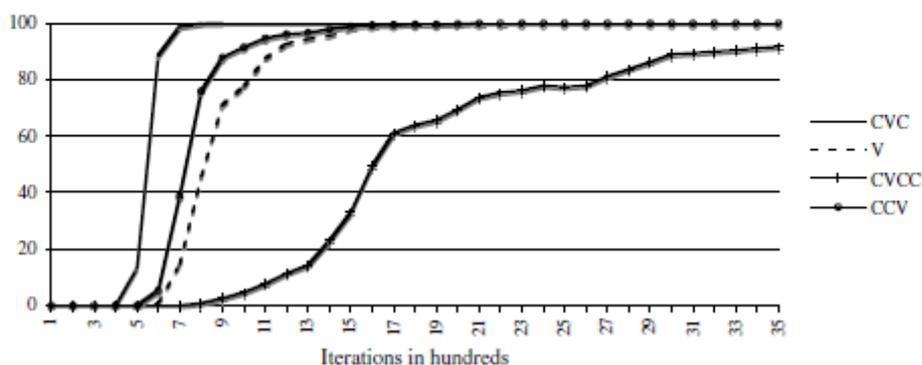
CV	CVC	CVCC	V	VC	VCC	CCV	CCVC	CCVCC
24.4%	40.5%	10.1%	4.7%	13.0%	3.5%	0.9%	2.2%	0.6%



Jarosz p. 598

TABLE 6. *Relative frequencies of syllable types in Polish*

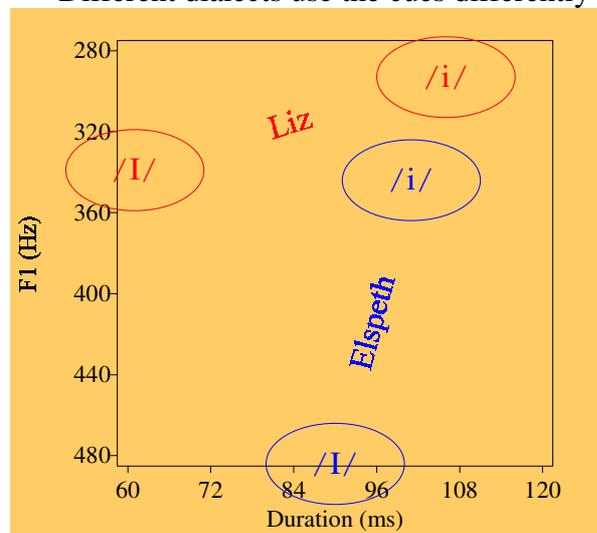
CV	CVC	CVCC	V	VC	VCC	CCV	CCVC	CCVCC
50.3%	20.9%	3.3%	8.5%	3.5%	0.6%	9.6%	4.0%	0.6%



Jarosz p. 600

9 Escudero & Boersma 2001: learning to weight perceptual cues

- English /i/ (“peach”) and /ɪ/ (“pitch”) are differentiated by two main cues
 - F1 (\approx tongue/jaw lowering) and duration
- Different dialects use the cues differently in production:



“Elspeth”: a Scottish English speaker
 “Liz”: a Southern British English speaker

Escudero & Boersma slide 6

- So the boundary between the two categories in this 2-dimensional space is something that must be learned

- Boersma has been a proponent of using OT tableaux for perception. Example:

[350 Hz, 80 ms]	350 Hz \neq /i/	350 Hz \neq /ɪ/	80 ms \neq /i/	80 ms \neq /ɪ/
perceive as /i/		*		*
perceive as /ɪ/	*		*	

This is too low to be [ɪ] in Southern dialect, too short (and high) to be [i] in Scottish dialect

- I didn’t attempt a full demo, but we can look at Escudero & Boersma’s results
 - training data for each dialect: typical realizations of [i] and [ɪ]
 - If learner’s current grammar miscategorizes this item, ranking values are adjusted
 - At the end, one can test how often each Hz-msec combination is categorized as each vowel
 - ==> I’ll step through Escudero & Boersma’s results graphs on-screen

10 Summary of today and last time

- One way to characterize variation is as a probability distribution over non-variable grammars (some of which can generate the same outputs)
- In the case of OT, there are some natural restrictions we might want to put on such distributions
 - Anttila’s partial ranking: assign constraints to strata, within which each ranking is equally probable
 - Stochastic OT: assign a ranking value (number) to each constraint. A ranking’s probability depends on how far from that ranking value each constraint would have to be in order to produce it.

Next time: Convergence problems with the GLA and possible solutions. In-class lab on probability distributions over rankings

References

- Anttila, Arto. 1997. Deriving variation from grammar. In Frans Hinskens, Roeland van Hout and Leo Wetzels (eds.), *Variation, Change and Phonological Theory*, Amsterdam, John Benjamins. pp. 35-68.
- Boersma, Paul & Clara Levelt. 2000. Gradual constraint-ranking learning algorithm predicts acquisition order. *Proceedings of Child Language Research Forum 30*, Stanford, California, pp. 229-237.
- Boersma, Paul & David Weenink. 2012. *Praat: doing phonetics by computer*. Software package, www.praat.org
- Escudero, Paola Escudero & Paul Boersma. 2001. Attested correlations between the perceptual development of L2 phonological contrasts and target-language production confirm the Gradual Learning Algorithm. Slides from 25th Penn Linguistics Colloquium, Philadelphia. www.fon.hum.uva.nl/paola/
- Hayes, Bruce and Adam Albright. 2006. Modeling productivity with the Gradual Learning Algorithm: the problem of accidentally exceptionless generalizations. In i, edited by Gisbert Fanselow, Caroline Fery, Matthias Schlesewsky and Ralf Vogel. Oxford: Oxford University Press.
- Hayes, Bruce, Bruce Tesar, and Kie Zuraw (2011) "OTSoft 2.3.1," software package, <http://www.linguistics.ucla.edu/people/hayes/otsoft/>
- Jarosz, Gaja. 2010. Implicational Markedness and Frequency in Constraint-Based Computational Models of Phonological Learning. *Journal of Child Language* 37(3), Special Issue on Computational models of child language learning, 565-606.
- Niyogi, Partha. 2006 *The computational nature of language learning and evolution*. Cambridge, MA: MIT Press.