Automated Collection and Analysis of Phonological Data

James Myers
National Chung Cheng University
Lngmyers@ccu.edu.tw

Though phonologists have much to gain from quantitative corpus analysis and well-designed judgment experiments, the required technical sophistication goes beyond traditional training. MiniCorp and MiniJudge are free, open-source software tools (www.ccunix.ccu.edu.tw/~lngproc/MiniGram.htm) that automate these methods in terms of familiar linguistic notions like minimal pairs and Optimality Theory (OT).

I illustrate the tools in a study on Mandarin phonotactics. Mandarin disfavors triphthongs with identical high vowels (/i/ and /u/), as in (1a) vs. (1b), consistent with the Obligatory Contour Principle (OCP). Yet there are morphemes that violate this constraint, as in (1c). Following Pater (to appear), the exceptions can be analyzed in OT with a faith constraint Faith_{Ex} lexically indexed to them, giving the ranking Faith_{Ex} >> OCP. But does the mere existence of exceptions undermine the OCP hypothesis? If they are too rare to threaten the OCP, are they too rare to support Faith_{Ex}? Even if both constraints are supported, how can their ranking be tested? Finally, does the OCP pattern in the corpus remain synchronically active?

A MiniCorp analysis starts by tagging items in an electronic dictionary in terms of the constraints they violate. The end result is a table like (2), showing how many items violate each combination of constraints. MiniCorp then uses poisson regression to test the statistical reliability of a user-specified OT grammar, rather than modeling acquisition as is done by other loglinear OT models (e.g., Hayes & Wilson, to appear). Thus if violations of the hypothesized constraints are sufficiently rare, their regression weights (w) should be significantly ($p < .05$) below zero. Both constraints pass this test (Faith_{Ex} w = -6.51, OCP w = -5.59). Since Cons_1 >> Cons_2 iff $|w_1| > |w_2|$ (Prince & Smolensky 1993/2004), MiniCorp tests a ranking hypothesis against the null hypothesis that the constraint weights are identical. In the present case, the weights turn out to be too close ($p = .25$) to support the claimed ranking.

MiniJudge was then used to test for the synchronic activity of the OCP in native speaker judgments. The OCP predicts an interaction between two binary factors: [+/-FirstU] (whether the first vowel is /u/ vs. /i/) and [+/-LastU]. Thus [+FirstU+LastU] (/uVu/) and [-FirstU-LastU] (/iVi/) syllables should be judged worse than the others (/uVi/, /iVu/). The familiar notion of minimally contrastive sets

makes it easy to create nonwords defined by these factors, as in (3). MiniJudge then helps in the creation of additional test sets by substituting components of the original set consistent with the experimental design. The present experiment used four sets.

Twenty native Mandarin speakers were tested (though resampling shows that fewer may provide sufficient statistical power). Informants gave quick yes/no judgments (guessing was allowed), which cumulatively across items and speakers reflect gradient acceptability. MiniJudge uses a powerful loglinear regression technique (generalized linear mixed effect modeling; Baayen, to appear) that can compute by-item and by-speaker analyses simultaneously. In this case, MiniJudge found the regression weights in (4). The significant negative interaction shows that OCP violations were rejected more often than accepted, as predicted.

A stronger test of the OCP, however, adds a new factor to MiniJudge's regression analysis, namely the number of lexical neighbors for each item, computed by MiniCorp. The results in (5) show that acceptance increased for items with more neighbors, but factoring out this neighborhood effect caused the OCP interaction to reduce to nonsignificance. This suggests that the judgment pattern in (4) may have been due to analogy, rather than an OCP constraint in the grammar.

This study demonstrates that MiniCorp and MiniJudge permit quantitative analyses of corpus and judgment data to be run within a linguist-friendly framework (for other examples, see Myers 2007a,b). Current versions use Javascript (MiniJudge also has a Java version) to create code for statistical analysis to be run in R (R Development Core Team), but future versions will be even easier to use (ideally, free-standing Java implementations). All of the code is open-source (GPL), and I welcome both collaborators and competitors.

Figures

(1) a. iau^{51} "want", iou^{214} "have", uai^{51} "outside", uei^{51} "for"

b. *uau, *Cuau, *uou, *Cuou, *iei, *Ciei, *Ciai [C = consonant]

c. iai^{35} "cliff"

(2)

Count	Faith_{Ex}	OCP
1338		
4		*
0	*	
0	*	*

(3)

	[+FirstU]	[-FirstU]
[+LastU]	tuou^{35}	tiou^{35}
[-LastU]	tuei^{35}	tiei^{35}

(4)

	FirstU	LastU	FirstU x LastU (OCP)
Weights	-0.36*	-0.13	-0.47*

*$p < 0.05$

(5)

	FirstU	LastU	FirstU x LastU (OCP)	Neighbors
Weights	0.34	-0.94*	-0.12	0.04*

*$p < 0.05$

References

Baayen, R. H. (to appear). Analyzing Linguistic Data. Cambridge University Press.

Hayes, B. and C. Wilson. (to appear). A maximum entropy model of phonotactics and phonotactic learning. Linguistic Inquiry.

Myers, J. (2007a). MiniJudge: Software for small-scale experimental syntax. International Journal of Computational Linguistics and Chinese Language Processing, 12 (2): 175-194. [http://www.ccunix.ccu.edu.tw/~lngproc/Myers-CLCLP.pdf]

Myers, J. (2007b). Testing phonological grammars with dictionary data. National Chung Cheng University ms. [http://www.ccunix.ccu.edu.tw/ ~lngproc/IWGE_MyersPhon.pdf]

Pater, J. (to appear). The locus of exceptionality: Morpheme-specific phonology as constraint indexation. In S. Parker, ed., Phonological argumentation. Equinox.

Prince, A. and P. Smolensky. (1993/2004). Optimality Theory. Blackwell.

R Development Core Team. (2007). R: A language and environment for statistical computing [Computer software]. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0. http://www.R-project.org.