Class 7: Weighted constraints I; lab

To do for tomorrow (Friday)

- Read section 4.7 of Martin 2007.
  - Reading question will be: Why does [adding the "Gaussian prior" to the expression that MaxEnt is maximizing] help avoid overfitting? (I added syntactic bracketing because that sentence is hard to parse)
  - Turn in a brief response to the question (1 page or shorter)

Overview: A different type of constraint interaction: weights. Variation in noisy Harmonic Grammar. Lab on Harmonic Grammar.

1 Reconciling preferences

- We can think of constraint interaction as **reconciling preferences** of different constraints
  - a very general problem in computer science, machine learning, decision theory...
  - even a problem in designing voting systems (with people instead of constraints)

- What is this tableau saying?

/tɛft/	*ɛ#	*VOICEDOBSTR	*COMPLEX	MAX-C	NOCODA
a tɛft			*		*
b tɛf				*	*
c tɛ	*			**	

  - *ɛ# prefers *a* and *b* over *c*
  - *VOICEDOBSTRUENT doesn't care about these candidates
  - *COMPLEX prefers *b* and *c* over *a*
  - MAX-C prefers *a* over *b* and *c*, and *b* over *c*
  - NOCODA prefers *c*

- Because these preferences **conflict**, we need a way figure out which ones matter more.

- In Classic OT, the higher-ranked constraints' preferences can't be overturned
  - lower-ranked constraints are relevant only to the candidates still "unordered"

/tɛft/	*ɛ#	*VOICEDOBSTR	*COMPLEX	MAX-C	NOCODA
a tɛft			*		*
b tɛf				*	*
c tɛ	*			**	
constraint's preferences	a b \ / c	a b c	b c \ / a	a | b | c	c / \ a b
building up an overall preference ordering	a b \ / c	*no change*	b | a | c	*no change*	*no change*

- But that's not the only way to do it!
- Other possibilities...
  - majority rules for top candidate: *a* gets ½ vote + 1/3 vote + 1 vote (but so does *c*0
  - majority rules for each pair: *b*>>*c* in 2/3 of the constraints that care
  - easier: give each constraint a number that says how much it matters

- Harmonic Grammar (Legendre, Miyata & Smolensky 1990): each candidate's score is the **weighted sum** of its violations

weight	-5	-3	-3	-2	-1	
/tɛft/	*ɛ#	*VOICEDOBSTR	*COMPLEX	MAX-C	NOCODA	*penalty score*
a tɛft			*		*	-3+(-1)=-4
b tɛf				*	*	-2+(-1)=-3
c tɛ	*			**		-5+(-2)+(-2)=-9

  - *b* wins because it has the best penalty score (closest to 0)
  - There's no variation here yet—regular Harmonic Grammar is not a theory of variation.

Harmonic Grammar is different from Classic OT

- Fill in the penalty scores and determine the winners. How is this different from classic OT?

weight	-3	-2	
/op/	NOCODA	MAX-C	*penalty score*
a op	*		
b o		*	
/isk/	NOCODA	MAX-C	*penalty score*
c isk	*		
d is	*	*	
e i		**	

- Same question:

weight	-3	-3	
/kaptol/	NOCODA	MAX-C	*penalty score*
c kap.tol	**		
d kap.to	*	*	
e ka.to		**	

See Keller & Asudeh 2002 for two real cases along these lines

2 Noisy HG: turning it into a theory of variation

- Each time you want to produce a tableau,
  - add some (normally distributed) noise to each weight
  - perform HG as usual to find the winner, using these perturbed weights

==> Demo in Excel file

3 Noisy HG and multi-site variation

- Remember on the first day when we talked about cases where there are multiple sites for variation within a word (Kaplan 2008, 2011; Vaux 2008)?
- Local optionality (see Day 1 handout for references): each site varies independently
- Do you remember the problem for this tableau?

/mɑɹkətəbɪlət**i**/	*t/V_V[1]	IDENT(continuant)
☞ *a* [mɑɹkət^{h}əbɪlət^{h}i]	**	
☞ *b* [mɑɹkərəbɪlərɨ]		**
☞ *c* [mɑɹkət^{h}əbɪlərɨ]	*	*
☞ *d* [mɑɹkərəbɪlət^{h}i]	*	*

[1] big simplification

- In Noisy HG, how could we assign weights to get this to work?

- Will Noisy HG work for global optionality too (all sites have to behave the same)?

/hapisapa/	*p	IDENT(voice)
☞ *a* hapisapa	**	
☞ *b* habisaba		**
c hapisaba	*	*
d habisapa	*	*

- How about iterative optionality?

/ɔ́ ká zā pī/	ALIGN(ATR)	IDENT(ATR)
☞ *a* ɔ́ ká zā pī	***	
☞ *b* ɔ́ ká zɐ̄ pī	**	*
☞ *c* ɔ́ kɐ́ zɐ̄ pī	*	**
☞ *d* ɔ̹́ kɐ́ zɐ̄ pī		***

- Single-site optionality?

/invitado/	*OPENSYLLABLE	DEP-C
a in.vi.ta.do	***	
☞ *b* in.vis.ta.do	**	*
☞ *c* in.vi.tas.do	**	*
☞ *d* in.vi.ta.dos	**	*
e in.vis.tas.do	*	**
f in.vis.ta.dos	*	**
g in.vi.tas.dos	*	**
h in.vis.tas.dos		***

4 Goodness scores

- Recall also from Day 1 that sometimes we need to know not just which candidate wins but how good it is, to use in modeling...
  - which new words get accepted in the language,
  - which derivational morpheme to use (-ian, -an, -er, -ese,... for 'person from-')
  - which lines of poetry get used
  - which names become popular
  - etc.
- Noisy HG (and regular HG too) provide each candidate with a goodness score!
- Coetzee & Pater 2008 use these scores to model how frequent words with different consonant combinations are in the Muna language.
  - Applying the idea to the English sCVC cases:

weight	-10	-1	-1	
/smæp/	IDENT(place)	OCP-LABIAL	NOCODA	*penalty score*
☞ *a* smæp		*	*	**-2**
b snæp	*		*	-11
c smæt	*		*	-11

weight	-10	-1	-1	
/smæk/	IDENT(place)	OCP-LABIAL	NOCODA	*penalty score*
☞ *a* smæk			*	**-1**
b snæk	*		*	-11
c smæt	*		*	-11

(see their paper for how they turn those scores into predicted type frequency)

	weight	-10	-1	
	/smæp/	IDENT(place)	OCP-LABIAL	*penalty score*
☞ *a*	smæp		*	-1
b	snæp	*		-10
c	smæt	*		-10

5 Learning algorithm for HG: Gradual Learning Algorithm!

- Boersma & Pater (to appear) prove that this procedure converges correctly when the data are non-varying:
  - Every time the current HG grammar (with or without noise) makes an error, increase the weight of the constraints that favor the winner and decrease the weight of the constraints that favor the loser
- As you read in Coetzee 2009, this procedure also seems to work well for variable data.

Lab: Harmonic Grammar and Noisy Harmonic Grammar

6 Use OT-Help to learn (non-noisy) HG weights

- Visit http://people.umass.edu/othelp/.
- Go to the download page and follow the instructions for downloading
- OT-Help uses OTSoft input files!
  - So, use any OTSoft input file that you already have, but you have to first save it as a *.txt file. (Try one *without* variation first.)
  - Drag and drop your *.txt input file into the OT-help window
  - Choose "parallel", not "serial" (unless you are familiar with Harmonic Serialism and want to try it)
  - Take a look at the "OT Solution" and "HG Solution" results

7 Use Praat to learn (non-noisy) HG weights

(Modified from a handout by Joe Pater: http://people.umass.edu/pater/praat-learning-guide-jp.pdf)

- Set-up:
  - Download and open Praat (www.praat.org)
  - Download the Praat-formatted version of the Anttila Finnish data (http://www.fon.hum.uva.nl/paul/gla/Anttila_data.txt)
  - In the `Praat objects` window, `Read > Read from file...` and select the Anttila data file
  - You'll see that three new objects now appear in your `Praat objects` window: the `OTGrammar` (constraints and tableaux), `PairDistribution` (the frequency of each candidate), `Distributions target` (normalized frequencies)
  - Select the `OTGrammar` object and click the `Edit` button to have a look at your grammar (before any learning has happened)
  - Select the `PairDistribution` object, and click the `To Table` button.
  - A new `Table` object will appear in your objects list. Select it and click the `Edit` button to see what it contains
  - Select the `Distributions` object, click `Convert - > To table...`. Click `OK`.
  - Another new `Table` object appears. Select it and click `Edit` to see what it contains.

- Learning
  - Select the `OTGrammar` object, click the `Modify behaviour` button, `Set decision strategy`, and choose "`Linear OT`". Click `OK`.
  - Select the `OTGrammar` *and* `PairDistribution` objects
  - You'll now see a button called `Learn`
  - You see a new window with various options—for now, leave the defaults as they are and click `OK`.
  - You'll see some learning progress happening in a new window, which closes when it's finished
  - After it's finished, it looks like nothing happened.
- Inspecting results
  - Select the `OTGrammar` object and click the `Edit` button.
  - A new window opens—you'll see that the constraint weights have changed.
  - In this new "`OTGrammar`" window, choose from the menus `Edit` > `evaluate (noise 2.0)`
    - This will simulate one trial of generation: adds some noise to each weight ("ranking value") then performs HG on each tableau with those noise-added weights
    - Try typing CTRL-2 repeatedly (the shortcut for `Edit > evaluate (noise 2.0)`) to see repeated evaluations
  - Close the "`OTGrammar`" window
  - Back in the `Praat Objects` window, select the `OTGrammar`. Click `To output distributions...` , accept defaults and click `OK`
  - Now a new `Distributions` object appears in your objects list. Select it and click `Convert > To table...` Click `OK`.
  - Select the new `Table` object and click `Edit`.
  - Compare the two `Table` objects to see how close the results are to the training frequencies.

8 Your own data

- [Sorry, I messed up this part]: Using the Anttila data or another data file from Boersma's page, such as Ilokano, convert the OTSoft input file with your own data into Praat format.
- Learn a noisy HG grammar for your data.
- How well do the results fit the training data, as compared to the Stochastic OT grammar you made on Tuesday using the GLA in OTSoft?

9 Cross-validation

- If you got as far as making your 10 testing and 10 training files in Tuesday's lab, you can use those same files for noisy HG learning
  - If you didn't get to that step on Tuesday, try it now
  - You'll need to convert the files into Praat format (you can use OT-Help)

Once again, save all your output files!! They will be useful for your presentation.

Next time: Our last quantitative constraint model, which also uses weights: Maximum Entropy OT. We'll use MaxEnt to revisit the questions of overfitting and smoothing—including from an empirical perspective (i.e., do human learners do smoothing?). Plus a MaxEnt lab.

References

Boersma, Paul and Joe Pater. To appear. Convergence properties of a gradual learning algorithm for Harmonic Grammar. In John McCarthy and Joe Pater, eds. *Harmonic Grammar and Harmonic Serialism*. London: Equinox Press.

Coetzee, Andries and Joe Pater. 2008. Weighted constraints and gradient restrictions on place co-occurrence in Muna and Arabic. *Natural Language and Linguistic Theory* 26: 289-337.

Coetzee, Andries W. & Joe Pater. 2008. Weighted constraints and gradient restrictions on place co-occurrence in Muna and Arabic. *Natural Language and Linguistic Theory* 26:289-337.

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

Kaplan, Aaron F. 2011. Variation Through Markedness Suppression. *Phonology* 28(03). 331-370.

Keller, Frank, and Ash Asudeh. 2002. Probabilistic learning algorithms and Optimality Theory. *Linguistic Inquiry* 33:225–244.

Legendre, Géraldine, Yoshiro Miyata, and Paul Smolensky. 1990. Can connectionism contribute to syntax? Harmonic Grammar, with an application. In *Proceedings of the 26th Regional Meeting of the Chicago Linguistic Society*. ed. by M. Ziolkowski, M. Noske, and K. Deaton, 237-252. Chicago: Chicago Linguistic Society.

Pater, Joe. 2008. Gradual Learning and Convergence. *Linguistic Inquiry*.

Vaux, Bert. 2008. Why the phonological component must be serial and rule-based.. In Bert Vaux & Andrew Nevins (eds.), *Rules, constraints, and phonological phenomena*. Oxford University Press.