Explorations for Pierrehumbert (2001)

Parameters of the Pierrehumbert model

Parameter	Description	Pierrehumbert value
τ	Decay rate of exemplars	2000
$\mathcal{E}$	Magnitude of uniform distribution from which the production noise ϵ is drawn	0.1
w	width of window used in classification	0.05
λ	lenition bias	varies
n_{trench}	number of exemplars to average over for entrenchment	500
p	probability of producing category 1, vs. category 2 (for Model 4)	0.75

Exploration topics

Your group should choose a topic from the list below, then post your choice to Piazza. Try to choose a topic which no other group has chosen. At most two groups may take the same topic. Topics are listed roughly in increasing order of complexity.

1. **Noise.** In the paper, noise is drawn from a uniform distribution on $[-\mathcal{E}, \mathcal{E}]$. What is the effect of changing $\mathcal{E}$ on the evolution of the mean and variance, for the one-category case?
2. **Entrenchment: two categories.** Explore the evolution of the two category distributions in a two-category model with entrenchment only ($\lambda = 0$). Do the category variances stabilize as for the single-category case? (Optional: What does the evolution look like when lenition is added ($\lambda > 0$)? How do the values of λ and n_{trench} interact in determining the evolution?)
3. **Window size.** How does changing the width of the classification window w impact the evolution in the two-category case, when other parameters (λ, τ, p) are held constant?
4. **Lenition.** What happens in the neutralization model as the amount of lenition (λ) is varied? Does the speed at which the categories merge change? Does the location of the peak of the merged category change?
5. **Changes to τ.** How does changing τ effect the evolution of the mean and variance of a single category over time? Explore and compare two or more of the one-category models (noise only, noise + lenition, noise + entrenchment, noise + lenition + entrenchment) for several different values of τ.

6. **Entrenchment: one category.** Examples in the paper and in class showed that the variance of a single category stabilizes with the implementation of entrenchment. Check if this is true for a range of values of n_{trench}. In addition, evaluate the claim that the size of the stable variance decreases as n_{trench} increases.

7. **Frequency and lenition.** The neutralization model presented in the paper includes entrenchment, a lenition bias for the 'marked' category, and a higher frequency ($p = 0.75$) for the marked category. Pierrehumbert suggests that the difference in category frequency and the lentiion bias drive neutralization of the marked category towards the unmarked category. However, as shown in class, no neutralization took place in a simulation without a lenition bias. Explore the consequences of varying p and λ. What is required for neutralization to obtain: differences in p, λ, or both?

   Advanced topics (optional)

8. **3 categories.** Modify the architecture to add a third category label. What kinds of behaviors do you observe as the probabilities of producing each category label are varied? Are these behaviors modulated by the inclusion/exclusion of lenition, entrenchment, or both?

9. **Alternative scoring function.** Instead of a square function, implement a scoring function where *all* stored exemplars contribute, weighted by similarity to the target, e.g.

$$f(x) = \sum_{i=1}^{N} w_i k(x, x_i)$$

   (This is actually a simple one-layer neural network.) How do the results compare to simulations using the window function used in the paper, when all other parameters are held constant?

Suggestions

- The number of values you try when varying a parameter is up to you, but you should make sure to report a range of values which demonstrate how the behavior of the model changes as the parameter is varied. (For example, $n_{trench} = 100, 101, 102$ will probably give very similar behavior when other parameters are held constant, while $n_{trench} = 5, 100, 250, 500$ will not.)
- You may wish to run your initial simulations for a relatively short number of iterations (e.g., 10000) to check that they are doing what you think they should be doing before running them for the full 100,000 simulations.
- Before modifying any parameters, find a model covered in class to start from. Check that you can run this model using the R functions provided and get results similar to those shown in class.