Modeling the Role of Social Information in Speech Perception

By

Emily Ada Remirez

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Linguistics

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Keith Johnson, Co-chair
Professor Kevin B. McGowan, Co-chair
Professor Gašper Beguš
Professor Isaac Bleaman

Spring 2024

Abstract

Modeling the Role of Social Information in Speech Perception

By

Emily Remirez

Doctor of Philosophy in Linguistics

University of California, Berkeley

Professor Keith Johnson, Co-chair

Professor Kevin B. McGowan, Co-chair

How do we model the relationship between “high level” social constructs and “low level” automatic processing of phonetic detail? Variation in pronunciation is socially informative, and listeners can draw on these social expectations when perceiving speech. This dissertation argues for a closer consideration of variation within sociophonetic exemplar modeling. I do this by reviewing the web of literature, simulating perception events in Python, and conducting an experiment. “Exemplar theory” is a class of models positing that past experiences interpreting stimuli are remembered as exemplars; new stimuli are categorized based on comparison to these stored memories. In particular, I focus on the Generalized Context Model (Nosofsky 1986; Johnson 1997), or GCM. The evidence that social categories, like other higher-order abstractions from stimuli, can play a role in categorization is well-established but loosely unified. Many adopt an episodic or exemplar-based framework in interpreting their results, but focus on the general patterns more than a specific model. I developed a Python library ExemPy which implements the GCM and provides routines for simulating common perception experiment tasks. I suggest applications for both enhancing empirical work and exploring theoretical space. I designed an experiment to explore a key difference among sociophonetic priming literature: whether social expectation is invoked as part of or outside of the phonetic stimulus. Taken together, this work advances an integrative, ecologically informed approach to exemplar-based sociophonetic research, drawing on multiple sources of evidence to contextualize our modeling.

Contents

Acknowledgments ii

Chapter 1: Introduction 1

Chapter 2: Episodic models and the GCM 5

Chapter 3: Sociophonetic priming 18

Chapter 4: The ExemPy library 25

Chapter 5: An experimental investigation of cue types in sociophonetic priming 45

Chapter 6: Conclusion 70

Bibliography 75

Appendix 1: ExemPy code and documentation 80

Appendix 2: ExemPy demonstrations 96

Appendix 3: Voice norming questionnaire 144

Acknowledgments

I saved this section for last. Mentors told me to write whatever I could, whenever I could. “Grab the low-hanging fruit,” they said, “do the citations or acknowledgments, things you know the answer to already.” But skipping straight to meditating on all the incredible people who’ve touched my academic career felt like cheating. It was a reward I hadn’t earned at the end of a quest I hadn’t completed. Well, here we are at last.

I am a grandparents’ girl first and foremost, so I’ll start with my grandmother Emma and late grandfather Mario. Their unconditional love and support have been an incredible constant and well of strength in my life. *Para mi Nana*, *gracias por todo, y te quiero más*. I’d also like to thank my parents, Sheri & Antonio, and my respective stepparents, Rick & Dana. No thank you to my big sister Andria for setting the bar so high, inspiring me to rise to the occasion out of a mix of pride and jealousy. I’m so lucky to come from a family that taught me curiosity, empathy, creativity, standing up for your convictions, and a healthy amount of whimsy.

I can’t fail to mention the family I’ve built in Berkeley. To my partner Oleg, my love: none of this would have been possible without you. You and our stinky, spotted little dog are my home. Oleg and Luna are just one part of a larger network here in California, including: My college roommate & dear friend who lived across the Bay, Victoria Eng, who provided so much fun and love and light, and always wanted to celebrate every small success along the way. My lab-mates Sarah Bakst, Alice Shen, Auburn Barron-Lutzross, & Matt Faytak, and my cohort-mates Julia Nee & Raksit Lau-Preechathammarach. The encouragement and companionship of our community has been such a gift. I wish all y’all that moved away would come back. Thank you to the rest of my Baker College pals for all the long-distance but persistent friendship (and casual gameplay).

This work would not have happened without the personal and professional guidance of my first graduate advisor, Dr. Susan Lin. To Susan: I hope you already know everything that your mentorship has meant to me. Thank you for seeing me for who I am, for pushing me to be my best with grace and compassion. To my other readers, I hope you know that Dr. Lin is one of the kindest, cleverest, goofiest people you could have the fortune to know. Her analyses are insightful, intuitive, and interconnected; she is, in a word, inspiring.

My advisors Keith Johnson and Kevin B. McGowan have likewise guided me with compassion, humor, and really smart ideas. They've both stepped up for me again and again. Thank you for everything. (I bet y'all are so happy this is done.) I won't write more, or it will embarrass them.

Thank you to my committee members Gašper Beguš and Isaac Bleaman for all their wisdom. My coding skills in general and the specific code in this dissertation owe a great debt to Ronald Sprouse and Terry Regier. They were both so incredibly generous with their time and confidence in me, experts in both their advice and the way they deliver it. Belén Flores, Andrew Garrett, Christine Beier, and Brian Wilson provided invaluable and sincerely appreciated moral support and academic guidance within the department.

Thank you to everyone who contributed to ExemPy and the experiment in Chapter 5. I'm grateful for collaboration with friends and colleagues Sarah Bakst, Raksit Lau-Preechathammarch, Matt Faytak, Auburn Barron-Lutzross, and Anna Björklund, which greatly helped develop ExemPy. The experiment couldn't have happened without undergraduate research assistant Jiacheng Liu and volunteers Sarah Miller and Lia Christine Dewey's help in creating the stimuli. Thank you to all of the research volunteers on Prolific and my norming survey who took the time to make the study possible.

Thank you to the fellow students I shared space with and learned from in the department and across campus. It was pleasure to spend my time with such curious and brilliant people. I'm proud to have participated in the historic Fall 2022 strike along with other academic workers across the University of California system. I'm grateful to the organizers and workers who came together to fight for a more fair workplace in which we can be better teachers and reesarchers. Thank you to the communities at the Gender Equity Resource Center, the American Cultures Engaged Scholars program, the Office for Graduate Diversity, and the Office of Undergraduate Research & Scholarships.

Working with inquisitive and eager students, witnessing their growth, and getting to take credit for some small part of their successes was *crucial* for keeping my own spirits high. Thank you to the undergrads who trusted me as a teacher and a mentor. My students, SURF & GiGS advisees, and LRAP apprentices have taught me so much. Thank y'all for reminding me so often what this is all about with your enthusiasm, optimism, and unique perspectives.

Before there were the people who got me through my PhD, there were incredible mentors at Rice University who set me up to get here. To name just a few: Robert Englebretson, Michel Achard, Matt Shibatani, Aysha Pollnitz, Mark

Gibson, and Elaine Howard Ecklund. (If Kevin McGowan, my first ever linguistics teacher, hadn't joined my committee, he would get to be here too.) Each taught me lessons too important and numerous to be summarized here. Here, I'll thank them for getting me through Rice feeling confident, supported, and prepared. To Robert, Dr. E: thank you for always having time for me, at Rice and after. Your nuanced and person-centered approach to language research is inspiring. Thank you for setting a gold star example for how to teach and show up for other people. I also need to mention my favorite study buddies Claire Elestwani, Dennis Budde, and Alec Weiss, who sustained my interest in the field through the power of friendship and group projects.

The public have invested state and federal funding into my research and education. Working for Professor Ecklund at Rice, I listened to and read a lot of interviews in which scientists discussed their research ethics. Many of them brought up an idea which has stuck with me ever since: Receiving public money creates an obligation to the public. Thank you to everyone who believes in investing in knowledge and education, and to every researcher who does their work with this commitment in mind. Thank you to public educators everywhere.

Finally, shout out to my haters.

Chapter 1
Introduction

Humans use language to build social structures that are too large to be viewed at anything other than a macro level, yet are instantiated in the most local of interactions. Any "high-level" construct is carried by numerous individual occurrences, each with their own "low-level" details (e.g., Mendoza-Denton 2014). This is true for linguistic categories like *voiceless* and *noun*, and it's true for social categories like *woman* and *cowboy*. Listeners draw on multiple "levels" of information during perception, including social categories. Those who expect to hear a certain type of speaker may perceive that person's speech as being more characteristic of that group. Because these effects can be induced, a robust model of perception needs to account for them.

So, how do we model the relationship between these high level social categories and the low level perception of speech? I contextualize this question within two overlapping strains of research: Exemplar modeling and sociophonetic perception of spoken language. I present both a review of the literature and an empirical contribution within each sphere. Specifically, I introduce and demonstrate ExemPy, an original Python library for simulating speech perception events and report the results of a listening experiment. The chapters are outlined in more detail in Section 1.1 below.

A recurrent theme in this dissertation is to ask what it means for things to be *the same*. How are differences blurred or accentuated according to category membership? As Whorf wrote, category divisions are both non-obvious and consequential:

> We dissect nature along lines laid down by our native languages. The categories and types that we isolate from the world of phenomena we do not find there because they stare every observer in the face; on the contrary, the world is presented in a kaleidoscopic flux of impressions which has to be organized by our minds—and this means largely by the linguistic system in our minds. (Whorf 1940)

Or, as my professor Terry Regier once put it during a class discussion: "It's all variation; some of it's just a bit clumpy." Categories, in these terms, are the well-motivated but violable boxes we draw around those clumps.

On one level, I raise the topic of priming and activation spreading during perception. Inferring something to be a member of a category creates an expectation that the something will have certain attributes. That is, we expect it to have things in common with the examples of the category we've seen before– things that are "the same." We'll review some of the evidence for this type of top-down effect in Chapter 3, and consider some mechanisms for it in Chapter 4. On another level, the way that we, as researchers, categorize theories or findings impacts the questions we ask and conclusions we draw. Chapter 2, for example, begins with a review of variation within the category of "Exemplar Theory."

This term, without contextualization of the class of episodic models of perception, implies a single theory. Difficulty arises when we try to reconcile the varied, contradictory assumptions represented across the class. Emphasizing what these frameworks have in common is crucial for collaborating and synthesizing ideas; remembering where they differ is crucial for responsibly contextualizing them. The experiment I report in Chapter 5 operationalizes some of the differences across methodologies.

And, my motivation to connect these topics reflects a deeply held belief that what we try to study in the lab is in some way meaningfully the same thing– language–that our colleagues in fields like linguistic anthropology describe. If we believe that the same language users who are constructing and perceiving high-level identities out in the world are coming into the phonetics lab as test subjects, we can get a more accurate picture of speech perception by incorporating concepts from interactional and ethnographic accounts. This bridge draws us closer to an integrative, ecologically-informed approach to exemplar-based speech perception research.

In writing this dissertation, I've adopted a less formal and more conversational style of writing. Everyone has the capacity to understand these concepts and should be able to access them. In a way, I'm writing in the voice I would take with my students, committed to presupposing that we are all full of potential and curiosity and capable of achieving our goals. Further, when we begin at the margins of our target audience, stating things in clear and basic terms, it edifies the positions for everyone. I can only hope that I've been successful in breaking these ideas down, and am grateful for my committee members' support of this goal.

1.1 Summary of chapters

First, Chapter 2: Episodic Models and the Generalized Context Model introduces exemplar-based, or episodic, models of language and perception. In particular, we focus on a specific implementation proposed by Nosofsky (1986, see also Medin & Schaffer 1978) and elaborated for phonetics by Johnson (1997). The Generalized

Context Model (GCM) is an example of “Exemplar Theory.” This term is often used to capture what this set of models have in common: an overarching focus on specific experiences (episodes) of the listener’s language use. While this level of abstraction is useful and appropriate for many contexts, it makes it difficult to make and assess specific predictions. As we move toward disambiguating individual models from each other, it becomes even more important to remember that “Exemplar Theory” is not a single theory at all. In addition to restating and clarifying the basics of the model, I identify two key mechanisms that may be leveraged in modeling sociophonetic perception: Stable effects in base activation weights, and resonance, a cyclical process of categorization.

Contextualizing these mechanisms first requires us to understand the behavior they’re meant to account for. The evidence for these effects is well-established but loosely unified. Chapter 3: Sociophonetic Priming reviews previous findings showing that social information can condition perceptual responses in priming experiments. Listeners who expect to hear a certain type of speaker are more likely to make a choice that reflects those expectations. Given this body of evidence, a model of speech perception must be able to account for the integration of “non-linguistic” social cues and phonetic details. The chapter discusses both experimental results and theoretical accounts for them. With the simulations presented in Chapter 4, I target alternate accounts using the GCM. To close the chapter, I call further attention to the variety of ways “social factors” are construed in these experiments, previewing the motivations for my investigation in Chapter 5.

With that foundation laid out, Chapter 4: ExemPy, introduces and demonstrates a library of routines in Python which implements the GCM. I argue that perceptual simulation can be leveraged in both theoretical and empirical applications. The precise control of parameters allows hypotheses to be generated and tested under different accounts, enabling systematic exploration of a theoretical space. Likewise, where empirical data does not exist, simulations can be used, cautiously, as a non-resource-intensive proxy. After introducing the library in general, I demonstrate its functions using data from Peterson and Barney (1952), which I used in developing the library. Finally, I turn to two applications of the library. I collaborate with Dr. Sarah Bakst to leverage ExemPy in analyzing data from an altered auditory feedback experiment she conducted. We partially replicate an experimental result by using ExemPy in place of a sub-experiment. The chapter ends with some demonstrations of how two mechanisms–priming and resonance–could integrate social and phonetic information during perception.

Chapter 5 describes a study on sociophonetic priming that uses different types of cues to talker identity and measured their effects on a vowel categorization task. The results in Chapter 3 are re-considered with an emphasis on variation in their methodologies: What constitutes “social factors” in these experiments? How is expectation induced? How is perception measured? The experiment focuses on this second dimension in a matched guise lexical classification task. I divide the set of cue types into two broad categories. That is, the social category can be cued outside of the auditory stimuli, such as with a photo or label, or carried on the phonetics of ta stimulus itself. This experiment operationalizes that difference, probing the quality of these two cue types and their interaction. A “valley girl” persona, associated with the California Vowel Shift, is invoked through visual primes and through enregistered phonetic features. Listeners heard a re-synthesized continuum spanning “bat” and “bot.” The number of “bat” responses was used to measure degree of association with the sound change associated with valley girls. While unable to answer the question about cue types, I report the unexpected findings: In the final iteration of the experiment, I found no effect of the visual prime. Further, counter expectation, creaky voice was associated with less perception of centralization than modal voicing.

Chapter 6 concludes.

Chapter 2
Episodic Models and the GCM

2.1 Episodic models of language

To say that a model of speech perception, or of language more generally, is exemplar-based is essentially synonymous with describing it as episodic (Goldinger 1998). Exemplar-based models of speech perception differ from other models in a few key ways. Where those models depend on abstract, pre-existing categories to structure input from the top down, exemplar models build categories up based on experience. These frameworks posit that the listener stores memories of past perception events as examples of categories. Exemplar models differ from accounts like Motor Theory (Mattingly et al. 1988) and Direct Realist Theory (Fowler 1986) in that the object of perception is generally assumed to be acoustic and not articulatory—though there is no reason, in principle, that exemplars couldn't have articulatory components (Johnson 1997). Like Direct Realist Theory and general auditory approaches (Diehl, Lotto, & Holt 2004) and in contrast with modularist perspectives such as Motor Theory, the perception of speech in "Phonetic Exemplar Theory" (Hay & Bresnan 2006) is general to cognition rather than specific to language.

Speech perception involves parsing complex, variable, multi-modal signals into discrete, meaningful linguistic categories. An exact definition of "category" is hard to pin down. Recall from Chapter 1, I introduced categories in the context of variation. I referred to categories as well-motivated but violable boundaries drawn around clusters of similar things. Figure 2.1 elaborates the conception of categories that will be most relevant for this dissertation.

Episodic models of speech perception posit that the language user categorizes novel input by comparing that input to their memories. These examples are referred to functionally interchangeably as exemplars, episodes, and traces. The stores of memories are often called exemplar clouds or exemplar distributions. Crucially, these models can account for the same phenomena often discussed in speech perception literature, while also extending the power. Prototype-like effects and categorical perception fall out naturally from the fact that the "best" members of the category will likely be well-represented and near the center of the distribution, therefore producing the most activation in aggregate (see Hintzman 1986 on schema abstraction).

As mentioned above, these are also a natural choice for processes involving variation in speech, because variation in the signal is “baked into” the mechanism, rather than being extra noise to be accounted for. Listeners don’t need to normalize, or strip away all acoustic differences in novel speech to access some “pure” and abstract form of that speech (Johnson 1997).

Exemplar-based frameworks are general to categorization (Kruschke 2008) and have been elaborated into particular models in distinct areas of linguistics. The Generalized Context Model, which was developed out of work with visual perception Some find it helpful to capture these differences as Phonetic Exemplar Theory and Syntactic Exemplar Theory (Hay & Bresnan 2006, Bod & Cochran 2007). Like the term Exemplar Theory itself, this grouping blurs differences between the distinct models within the categories. And, it usefully captures the differences between how these theories have developed.

In syntax, perhaps the most famous example of an episodic model is Construction Grammar (CxG) (e.g., Goldberg 2006). In CxG and other constructionist approaches, morphosyntactic knowledge emerges from discourse in chunks called constructions, which vary both in size and level of abstraction. At the core, a construction is a pair between form and meaning. Although this mirrors almost exactly the definition of morpheme I give to undergraduate students, constructions can take on numerous different forms: Every word is itself a construction, but *transitive clause* and *polar question* are also constructions that take the form of templates with certain restrictions on what can fill in the slots (e.g., [animate noun] [transitive verb] [noun]). On the other extreme, idioms can be sentence-length constructions in which every aspect is fixed. When extended, shortened, or otherwise used creatively, we understand this as a reference to the construction. For example, many English speakers will be familiar with the phrase “too many cooks spoil the broth.” (When too many opinions are involved, the product can actually come out worse.) We can also refer to “too many cooks (in the kitchen)” or make playful versions like “too many advisors could spoil the dissertation.” That we’re able to subvert the construction this way can be taken as evidence that we have a shared understanding of these sentence-length constructions as units.

As Bod and Cochran (2007) state, “Phonetic Exemplar Theory” like the GCM, and “Syntactic Exemplar Theory” like CxG have developed largely independently. There is still behavioral evidence that supports their joint predictions. For example, Hay and Bresnan (2006) find that the degree of advancement in a New Zealand English sound change in the words “give” and "hand" vary systematically based on the construction type. That is, the phrase-level construction "giving a hand" is pronounced differently than the word-level constructions that comprise it. Gahl and Garnsey (2004) don't frame their investigation specifically as a joint prediction of exemplar theories, but they do find higher rates of phonological processes associated with phonetic reduction in environments where syntactic structure is contextually predictable.

Goldinger observes that a hybrid model, which contains both experiential details and abstractions "may prove necessary to accommodate many linguistic processes" (1998). That is, given the complexity of human speech and communication, we may not be able to model everything with a purely episodic model without room for some of these "higher level" cognitive constructs. Bod writes that exemplars are not tokens of experiences but analyses of tokens (2006). The ability to carry out such an analysis, as well as the category labels used by the GCM, seem to imply some sort of abstraction at a very basic level. It's also unsettled what the unit of experience should be: lexemes, morphemes, phones, phrases, or even "discourse-sized chunks" as Goldinger suggests (1998). There are logical reasons to believe that any of these must be the basic unit. And, as Goldinger and Azuma (2003) argue, we have reason to believe that the "basic" unit is flexible, responding to the task. They show that even something as innocuous as the beliefs of stimulus talkers or research assistants regarding the experiment's hypothesis is enough to produce self-confirming results. Following this line of reasoning, perhaps the reason we haven't isolated a single basic unit for exemplars is that there is not one basic unit. Whatever the case may be, Johnson (2006) writes that exemplars are incredibly rich in detail and are a type of recognition memory, not declarative memory. That is, even if speakers wouldn't be able to list out all of their experiences, they store multimodal, detailed representations of these experiences that they have subconscious access to.

To summarize, exemplar-based or episodic theories are general to cognition and categorization, and show up differently in different models, subfields, and domains (Hay & Bresnan 2006). In spoken language perception, these theories tend to differ from other models in a few key ways. I identify 3 unifying generalizations among exemplar models:

1. Humans store memories of past linguistic experiences
2. These exemplars are linked to categories
3. New stimuli are categorized based on comparison to the exemplars

2.2 The Generalized Context Model

Recall that “Exemplar Theory” is a *class* of models, which show a lot of variation. Just like any category, grouping these models together blurs a lot of the important differences and predictions. The Generalized Context Model, or GCM, is an extension of Medin and Schaffer’s (1978) context theory of classification (Nosofsky 1986). Although I use it here to understand speech perception, this model isn’t specific to language and was in fact developed in the visual domain.

Basics

The GCM can be summarized using the following equations. I’ve taken these particular formalizations from Johnson (2006), with only slight notational deviations. By comparing a stimulus *i* to past examples *j*, we calculate the probability that a language user would categorize the stimulus *i* as a member of a category *J*. The variable *j* is a stand-in for each stored example we’ll be comparing the stimulus *i* to. Each exemplar *j* was identified as belonging to various categories *J* during its perception and storage.

From the perspective of implementation, it’s useful to refer to a category list: a set of category types, each having a set of category labels. This relationship is visualized as a hierarchy in Figure 2.1; note this reflects the structure of the model implementation, not real-world categorization. The category labels, J are organized into category types, C. We assume that the category labels within a category type are mutually exclusive, and that these labels describe the same type of thing. In Figure 2.1, the objects can be categorized by two different category types: color and shape. The shape will be either category label “circle” or “square.” The dotted lines represent that a particular exemplar is linked to a category label under each category type.

To arrive at the probability that a stimulus is an exemplar of a particular category label, we need to calculate a series of values for each exemplar with respect to the stimulus: distance, similarity, and activation. These values are then totaled, separated by category labels. This gives us the evidence that the stimulus is a member of each category.

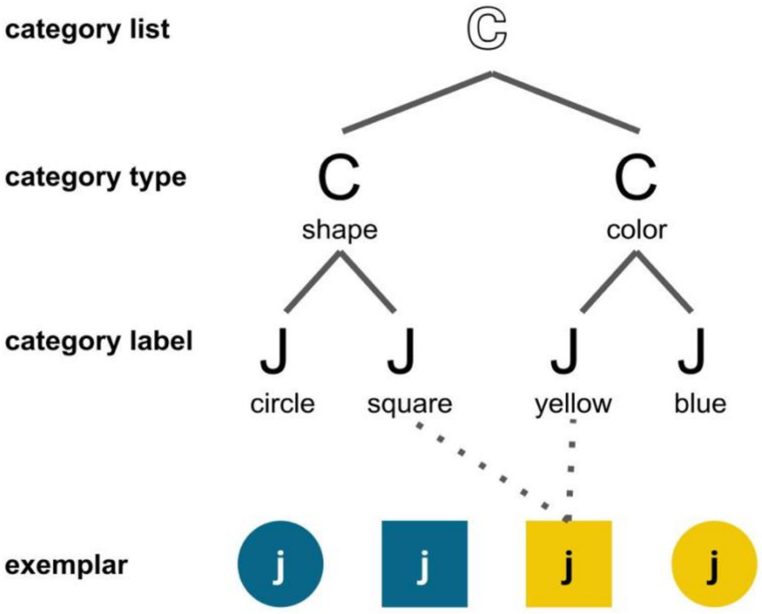


Figure 2.1: Schematization of the relationship between category types, category labels, and exemplars. The solid lines represent the hierarchical relationship between category types and labels. The dotted line represents the category labels assigned to a particular exemplar.

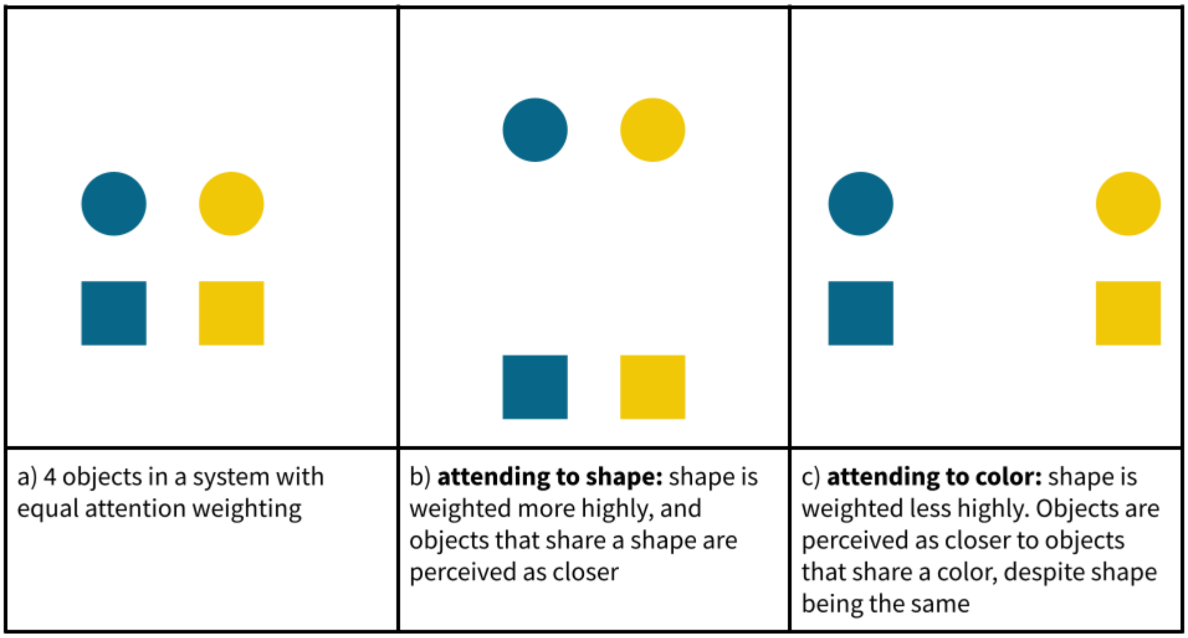
The distance between two exemplars is calculated based on the values *x* for each feature *f*, and a set of attention weights *w* for each feature (Equation 2.1). Elsewhere, you may see these features referred to as dimensions of variation *m*. This denotes any aspect along which something could vary, which therefore could be used to compare those things to each other. The term *x_if*[1] refers to the input's value for a given feature, and *x_jf* is the exemplar's value for the same feature. We calculate the squared difference between exemplars *i* and *j* for every feature and sum those differences up.

$$(2.1) \quad d_{ij} = \sqrt{\sum_f w_f (x_{if} - x_{jf})^2}$$

Distance is scaled according to how much "attention" is being paid to each feature, often normalized to sum to 1. Attention weights, *w*, scale the psychological distance between objects based on which features they have in common. It may be easiest to understand this visually. In Figure 2.2, consider wavelength (color) and number of sides (shape) to be two different features, or

[1] For convenience, I use underscores as notation for subscripts in the text.

dimensions of variation, we can use to describe each object. Imagine you’ve been asked to sort these objects into groups, or categories. The answer will change depending on how features are weighted. The more weight is given to a feature, the more similar objects with smaller differences in that feature will seem.



a) 4 objects in a system with equal attention weighting

b) **attending to shape:** shape is weighted more highly, and objects that share a shape are perceived as closer

c) **attending to color:** shape is weighted less highly. Objects are perceived as closer to objects that share a color, despite shape being the same

Figure 2.2: Feature weighting scales distance between objects.

The distance value calculated in Equation 2.1 is used to calculate similarity *S* (Equation 2.2). There are three components of this equation to pay attention to: Exemplar sensitivity *c*, the exponential *e,* and the negative value of the exponent -*cd_ij.*

$$(2.2) \quad S_{ij} = e^{-cd_{ij}}$$

Imagine that each exemplar has a field around it. If the stimulus fits within that field, that exemplar will become activated. Taking the exponential of the distance has the effect of distributing distances normally. We can look to Figure 2.3 to understand the negative value of the exponent: a larger sensitivity value produces a lower similarity. With a lower *c* value, exemplars with higher distances still correspond to some degree of similarity. However, when *c* is high, only those exemplars with low distance are considered similar and therefore contribute more activation. Because of the non-linear relationship between distance and similarity, the differences in similarity between any two distances will be larger at a higher *c* value; at a low *c,* these differences have less impact on similarity. That is, when *c* is high, a stimulus must be more similar to a given exemplar in order for that exemplar to be meaningfully activated. Following past research, *c* is treated as a constant. This is not done as a matter of proposition, but to constrain the endeavor and focus on other aspects. Unlike *w*, *c* cannot vary across dimensions, because it applies after the distances along each dimension have been summed to calculate distance *d*. However, it remains possible that *c* varies across individuals or even across tasks.

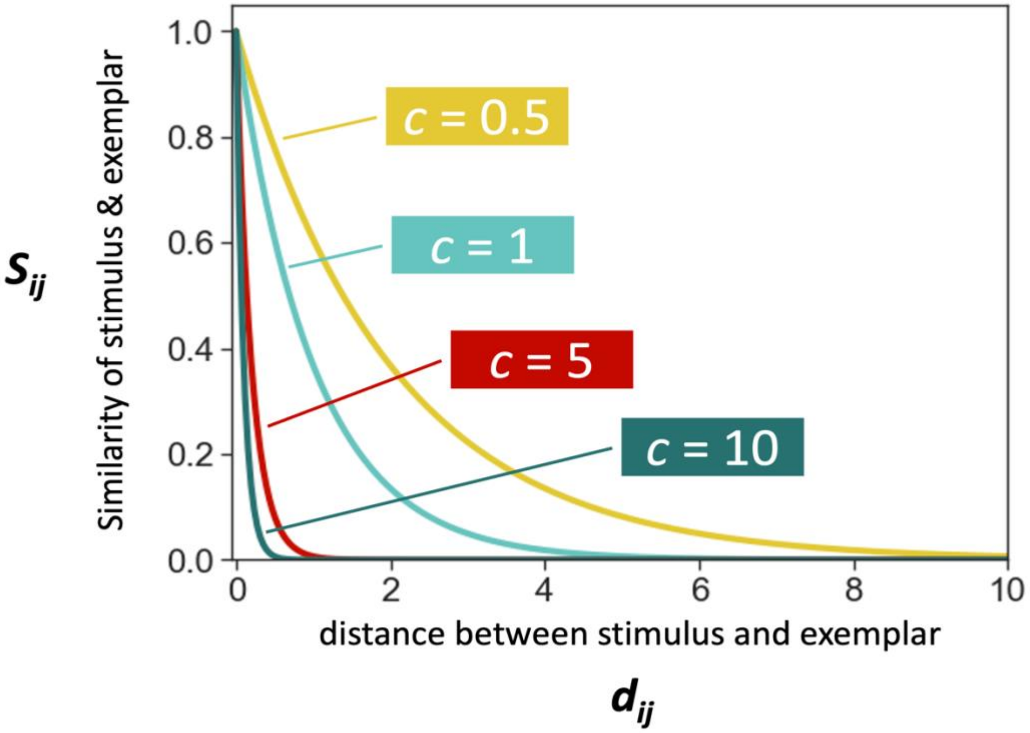


Figure 2.3: Similarity of i,j at different distances (0-10), with different sensitivity values (0.5, 1, 5, 10).

We use the similarity between the stimulus and each exemplar to calculate the exemplar activations *a* (Equation 2.3). Each exemplar is weighted by a starting activation value, *N_j*. In other words, when *N* is less than 1, activation will be a proportion of similarity *S*. When *N* is larger than 1, activation will be a multiple of *S*. This value represents a listener's expectations about what the stimulus will be. We attribute this value to properties such as the overall frequency of the category label and its likelihood within a specific context (Johnson 2006). For example, listeners have a tendency, known as the Ganong effect, to perceive auditory stimuli as real words even if the stimulus is not a real word (Ganong 1980).

(2.3) $a_{ij} = N_j S_{ij}$

To get the evidence that the input is a member of some category label big-J, *E_ J,i*, we add up the activation of every exemplar little-j that belongs to that label. We do this for every category label within each category type.

(2.4) $E_{J,i} = \Sigma a_{ij}, j \in C_J$

The equation in (2.5) is specialized from a more general axiom known as Luce's choice rule. Luce's choice rule normalizes the evidence relative to the total amount of activation in the system. The probability that A commonly cited property of exemplar theories is that more frequent categories have a bigger effect on perception. Frequency increases the chances to add activation, producing more evidence for that label. Because the increase in numerator is scaled by the increase to the denominator, frequency alone doesn't overwhelm the effect of similarity. Effectively, a more frequent label needs to be less similar to the input than labels with fewer examples. However, distance and similarity are still the basis of categorization.

(2.5) $P(R_J|S_i) = \frac{E_J}{\Sigma a_j}$

N and *N*-accretion

Base activation of an exemplar, *N*, scales the total activation of an exemplar, as in Equation 2.3. The higher the activation value of exemplar little-J, the more evidence that the stimulus belongs to category label big-J. When a social expectation has been induced, exemplars that share a category label are primed, raising their base activation level.

So, *N* can be selectively raised depending on consistent cues in individual speech perception events. But where does *N* come from in other contexts? I'd like to draw a parallel here between this kind of phonetic perception at what I consider a "low level" and some "higher level" phenomena that have been observed in discourse and ethnography. As Englebretson describes in the introduction to his 2007 book *Stancetaking in Discourse*, the term stance is used to encompass a broad range of phenomena, both in everyday speech and in research. In this same volume, Du Bois proposes a particular configuration of stance-taking, the stance triangle, defining stance as a social subject's simultaneous evaluation of objects, positioning of subjects, and alignment with other subjects (Du Bois 2007). We can understand stance as something highly local to specific interactions. Stance is performed through talk and interaction.

And yet, as Bucholtz and Hall (2005) note, citing Du Bois (2002), someone who is seen as repeatedly taking a particular stance—say, anger—in specific interactions can over time become known as an "angry" person. The "angriness" could then increase in indexical order (Silverstein 2003) and even come to be seen as an attribute of a group to which that person belongs or is associated. As with many enregistered cues or attributes, it often won't matter whether a stereotype is true of any one individual, or even accurate of the group as a whole. As Clopper's (2017) paper argues, stereotypes can fill a function similar to actual familiarity in perception experiments. Englebretson (2023) associates stance accretion with "third wave" models of sociolinguistic variation that emphasize speaker agency and context.

Stance accretion describes a mechanism in which something highly local and inherently dialogic can be inferred and constructed without interaction (Du Bois 2002, Bucholtz & Hall 2006). Through repeated use, the local performance of stance becomes part of a stable, transportable identity.

Stance performance is specific to the interaction. We can conceptualize individual speech events as something similarly local. By the same logic as stance accretion, can a particularly high base activation *N* of particular exemplars accrete through repeated occurrences? Let's say that something repeatedly raises *N* during perception of what Sumner et al. (2014) describe as "socially salient" speech. Perhaps through this repeated activation, the increase in *N* could accrete into something more stable, creating the reliable effect of variety observed in differences in encoding accounts. That is, we see robust representation of these salient varieties and variants not because they are encoded more strongly, but because their consistently high activation after the fact, during categorization, is "remembered" and allows them to play a larger role in future perception events.

This idea raises the possibility that there may be value in modeling two separate components to *N*. Again, let's draw an analogy to terms used in discourse. In Zimmerman's (1998) conception, a transportable identity is one that language users "carry around," such as their ethnicity or religious affiliation. It doesn't easily change, it isn't constructed by the discourse or the situation, but it can be invoked and become locally relevant. The transportable identity contrasts with positions based on institutional relationships (student/teacher, caregiver/child) or roles within the conversation (question asker/responder).

Despite their differences in origin, and in particular in "permanence," all of these identities can be relevant in discourse. It may be useful to consider what it would mean to split *N* into multiple components: one for more transportable effects, and one for more local. For the purpose of this argument, I'm committing to the assumption that any of these effects are subject to change and built up from individual examples. This is, after all, the primary difference I've noted between the GCM and difference in encoding accounts.

It may require a large effect to change the transportable base activation, while the local may be more violable. This is because local context changes very quickly, perhaps even from trial to trial of an experiment. The metaphor that keeps coming to mind involves the computer game The Sims, or at least my memory of it from my childhood in the early 2000s. Players operate simulated characters, sims, who interact and develop relationships with each other. The relationship between two sims is represented by two progress bars: a long-term relationship and a short-term one. The short-term meter fluctuates easily during an interaction. The long-term meter changes slowly as extreme positive or negative values are sustained.

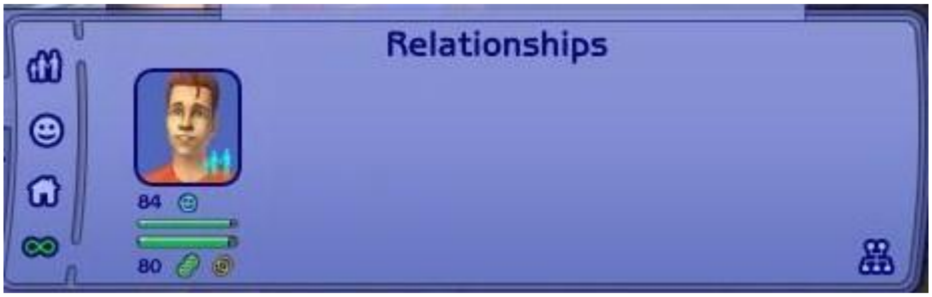


Figure 2.4: A screenshot from the Sims 2 computer game shows an icon of a character with two bars. The upper, long-term bar shows a score of 84, while the lower, short-term bar shows 80. Source: https://strategywiki.org/wiki/The_Sims_2/Tutorial_2

An alternative explanation is to seriously theorize hysteresis and some kind of elastic memory for *N*. It may be the case that the more an exemplar is primed, the more primeable it becomes.

Considering what to do with *N* raises an obstacle I've encountered repeatedly in this work. While we might understand what should happen at an algorithmic level, it becomes very difficult to choose particular values or arithmetic operations in a principled and consistent way. For example, it's easy to say that "*N* increases or decreases," and this intuitively makes sense. But how does it change? Is it multiplied, or added and subtracted? Do social expectations and other types of context (word frequency, syntactic predictability, Ganong effects, etc.) affect N at the same point in time? Are types of expectations scaled or weighted differently? This is one reason I argue that future work should remain skeptical about particulars of any implementation.

In service of unifying the differences in encoding approaches with the GCM, consider that strength in encoding, even without *N*-accretion, is exemplar storage with a particularly high *N*. This would counter the GCM's assumption that all exemplars are encoded with the same strength at the time of perception, but it would allow us to retain the flat structure of the exemplar cloud.

2.3 Resonance

Resonance is a cyclical process of categorization in which initial impressions re-enter the categorization process, spreading activation among related exemplars. This mechanism can integrate "macro" and local top-down expectation with bottom-up processing of linguistic and non-linguistic cues. The process of resonance is schematized in Figure 2.5

Strand (2000) finds that listeners respond faster to voices that are stereotypical for the gender categories "male" and "female," as judged by a separate group of listeners. The perception of sociolinguistic cues related to gender also vary according to the gender-stereotypicality of the voice. Johnson (2006) uses this finding to explore resonance.

In an exemplar resonance model, evidence for category membership accumulates over the course of exemplar activation (Johnson 2006). In an initial round of categorization, activation is calculated for each exemplar. This activation feeds back "up" to the category level. If there's a lot of evidence that the stimulus is a member of a category, activation spreads back down to other members of that category. If there isn't evidence, activation for those exemplars is shut off. This has the effect of blurring differences between members of some category. For me, it's easiest to think about this when, as in the case of gendered clustering in speech, a listener may use cues to make a decision about the speaker's gender. The bottom-up perceptual cues (phonetic or otherwise) create the same type of top-down expectation induced in sociophonetic priming experiments. Based on a first impression of a speaker, I make a decision and use that decision to structure subsequent categorization.

Johnson's description of resonance is related to Goldinger's (1998) notion of an echo (2006). Goldinger uses Hintzman's MINERVA 2 for demonstrating this concept, though he notes that the principle could apply to any exemplar-based model. During spoken word recognition, an analog probe P, analogous to our stimulus *i* goes out to all traces T (exemplars little-j). In a first round of activation, P activates each T proportional to its similarity, as in the GCM. An echo, which can contain both linguistic and non-linguistic information, is sent from long-term memory to working memory. Echoes are aggregates of activated traces and the total activity created by the probe. The content of the echo, then, is the net response to the probe. When this response is fed back into the categorization, it reinforces perception of features associated with that original percept.

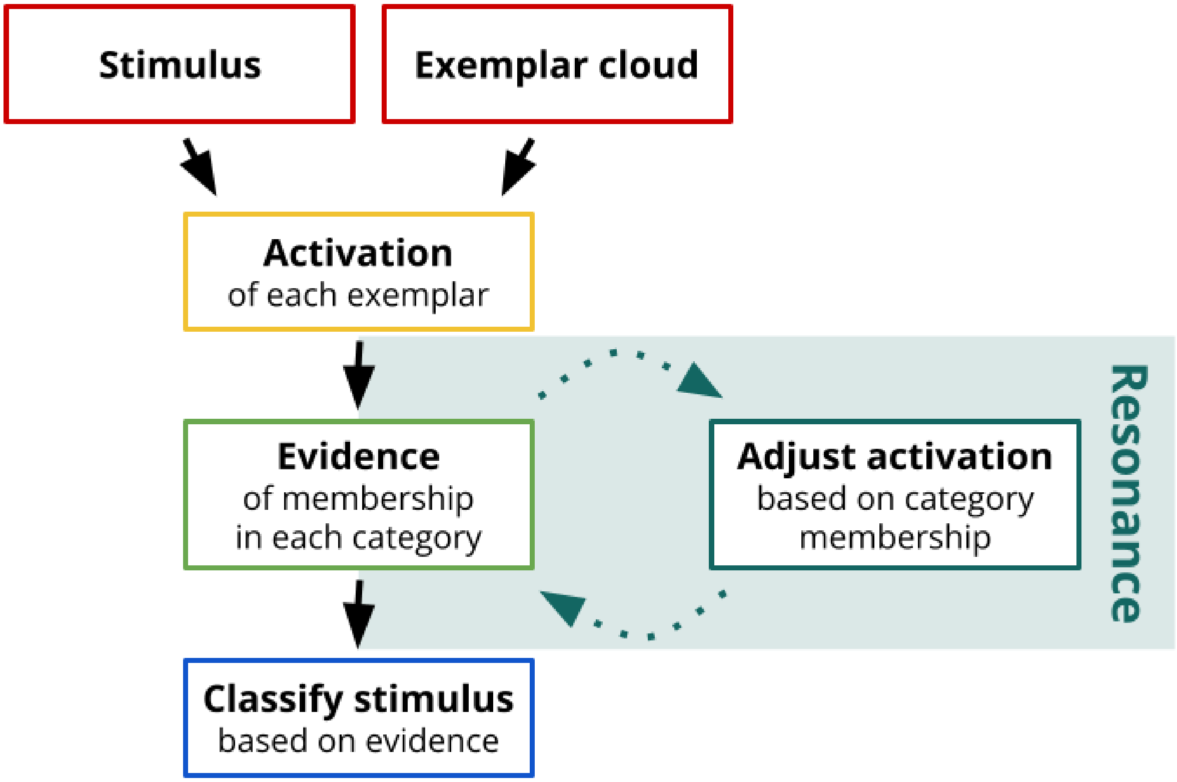


Figure 2.5: Schematization of resonance. The stimulus activates each exemplar within the cloud. That activation adds up to evidence within each category membership. In a unidirectional model, stimulus classification would be based on this evidence alone. In a resonance model, activation is adjusted based on this evidence, and evidence is recalculated.

2.4 Sociolinguistic Exemplar Theory

In the next chapter, I present some of the evidence used to argue for sociophonetic priming in speech. Sociolinguists may be particularly drawn to “Exemplar Theory” as a unifying model that encompasses many aspects of the linguistic and social system (Mendoza-Denton 2007). By considering linguistic knowledge as holistic and contextual, these approaches can account for both grammatical and social aspects within a single system. As Foulkes and Docherty (2006) point out, Exemplar Theory is uniquely suited for foregrounding socio-indexical information because “associations are automatically created in memory between linguistic and indexical information conveyed by the speech signal.” The characterization of “Exemplar Theory” as a coherent approach can be misleading, as it implies a compatibility among specific models that doesn’t in fact exist. For example, Sumner et al. propose dual-route encoding, wherein some exemplars are stored more robustly at the time of encoding (2014), while the GCM accomplishes this effect through activation weighting. Hay et al. (2006) suggest that non-useful aspects of exemplars decay over time, while Munson and Solomon (2004) suggest that listeners selectively store or not store entire exemplars. One reason I chose to look at the GCM in particular is a difficulty in reconciling aspects of “the same” theory.

2.5 Conclusion

In this chapter, we reviewed the class of models known as episodic, or “Exemplar Theory,” with a particular focus on the Generalized Context Model. I also introduced the idea that, analogous to theories in interactional work, stable effects in activation level may have origins in more mundane and regular interaction. Finally, we considered cyclical categorization, or resonance, as a bridge between bottom-up and top-down effects in perception. With this foundation, Chapter 3 will discuss literature supporting a particular type of behavior in perception: sociophonetic priming.

Chapter 3
Sociophonetic Priming

Some of the earliest evidence used to argue that social expectation guides speech perception comes from Nancy Niedzielski's 1999 study of listeners in Detroit, Michigan. Listeners associated a raised variant of the MOUTH diphthong with Canadian English, despite its being present in both regional varieties. Although the stimuli were identical, listeners who believed they were listening to a Canadian matched the stimulus to a more distinctive token with a more raised vowel nucleus than those who believed they were listening to a Michigander. Niedzielski interpreted her result by positing a "stereotype filter" that colors listeners' perception. Their stereotypes prevent them from hearing certain characteristics of speech. However, an episodic model of perception could also provide a parsimonious explanation. In their previous experiences, listeners could have paid less attention to "non-standard" features of their own community's speech, such as Canadian raising. The expectations set up by the task, combined with their attention to certain features, produced this difference in perception based on a social expectation.

The basic idea behind Niedzielski's investigation has been replicated in other contexts. The KIT vowel is more raised in Australian English (AusE) and more centralized in New Zealand English (NZE) relative to other varieties of English (Watson, Harrington, & Evans 1998, via Hay, Nolan & Drager 2006). Hay, Nolan, and Drager (2006) played New Zealand listeners words with the KIT vowel embedded either in the middle or at the end of sentences. Listeners recorded their responses on answer sheets that were labeled with either Australian or New Zealander. Those with Australian answer sheets matched the stimulus to a choice with a more raised KIT vowel ("feesh" for *fish*), while those with New Zealander answer sheets chose a more centralized vowel ("fush"). The authors theorize that the regional label primes exemplars of speakers from that area, making a more socially stereotypical pronunciation more highly activated.

In addition to labeling, analogous results can be found for facial stimuli. This follows naturally from the assumption that exemplars are rich, detailed recognition memories, as posited by Johnson (2006). For sighted language users, linguistic experience likely includes visual stimuli, relevant both for visual articulatory cues, such as labialization, and social identities conveyed by physical appearance, such as gender expression or ethnic phenotype. Both gender and ethnicity have been explored by Strand and Johnson (e.g., Strand & Johnson 1999, Strand 1999, Johnson 2006) and McGowan (2015). In Strand's work with stereotypical and non-stereotypical male and female voices, any voice paired with a stereotypically female face causes the boundary between sibilants /s/ and /ʃ/ to shift up in frequency, consistent with both a shorter vocal tract and performance of more feminine gender. McGowan (2015) found that the accuracy of perceiving speech in noise is improved when facial stimuli provide a social cue consistent with auditory stimuli. Listeners with both low and high exposure to Chinese-accented English listened to Chinese-accented English in noise, paired with either an Asian face, a white face, or a silhouette, and transcribed their speech. The transcriptions were most accurate with the Asian face, which is the most likely of the three to have been experienced along with Chinese-accented English. This work suggests that in laboratory settings, listeners use social information they infer from the putative speaker's face to influence their perception. By hypothesis, they do this in ways that have proven facilitative in their interactions with real speakers. That they employ this strategy, integrating various sources of information, in experiments hints that this could be a basic feature of speech perception outside the laboratory.

Each of these investigations support a model of speech perception which is rich, multimodal, and guided by our experiences. Our perception of speech is influenced by our expectations around the talker. In the next section, I further examine different types of social primes that have been linked to differences in perception.

3.1 What are "social factors" anyway?

As we saw above, there are several ways to deliver "social information," but what exactly do we mean by this term? I interpret the word "social" to mean anything dealing with how speakers position themselves or are positioned by observers with respect to other people, ideas, institutions, and objects. This includes a person's ethnicity, religious beliefs, gender expression, hobbies, sexual identity, language background, and more. Although many authors work with "social cues" or "indexical information," there seems to be nuanced differences in the way the term "social" is applied. While some authors stick to macro-demographic categories such as nationality, ethnicity, gender, or age-group (e.g., Hay, Nolan, & Drager 2006; Kim & Drager 2017; Hay & Drager 2010, McGowan 2015), others delve deeper into the nuances of identity construction and look at personae (D'Onofrio 2015, 2019). Some authors don't focus on separating out the specific social cues at all (e.g., Munson 2010, King & Sumner 2015, Kapnoula & Samuel 2019) and treat variation at a more holistic (but therefore less generalizable) level. The effect of social expectation holds both if the only cue comes from the voice itself (e.g., Strand & Johnson 1996, Johnson 2006), or if the social identity is cued at an abstract level (Hay, Walker, Sanchez, & Thompson 2015). In the rest of this section, we will review some of this literature in greater detail.

Hay and Drager (2010) extended the NZE vs AusE priming discussed above to an even more subtle class of social cue: the physical presence of either a koala or a kiwi stuffed animal in the lab. The koala is iconic oDf Australia, and the presence of this animal indeed leads to a more raised, Australian-like variant being chosen by participants. The kiwi, on the other hand, which is associated with New Zealand, is linked with the perception of more centralized, New Zealand-like variants. What's more, Hay, Drager, and Warren (2010) show that the native dialect of the experimenter who describes the instructions can also influence perception. When the experimenter was a speaker of RP, responses in a written, non-auditory oddball task were more RP like; when the experimenter was a speaker of North American English, the responses were more like those pronunciations. All of this research suggests that the social cues can be quite subtle, and possibly under the level of consciousness.

Hay et al. (2015) show additional support for abstract cues. In a combined implicit association task and written lexical decision task, they find an effect of association between gender-skewed words—words that tend to be used by one gender more than others, but do not explicitly refer to gender—and not only gendered faces, but gendered objects such as shoes. Although this effect is not the most robust or consistent, manifesting alternately in either reaction time or in accuracy, it extends even to objects which are only learned to be gendered in the context of the experiment. It is important to note that some of this work on NZE finds an effect of the listener's socio-economic background. As the authors noted, speakers with more financial resources are not only more likely to have had the opportunity to travel to Australia or other places where different varieties of English are used. The sound changes being investigated are, like many others, stratified by class within New Zealand. This suggests that by focusing on nationality, we may be obscuring some of the relevant intersections of this identity with others.

D'Onofrio's (2015, 2019) manipulation of *persona* adds some of this necessary nuance to the conception. Using broad macro-demographic labels such as race and gender has been quite common and fruitful in experimental settings.

However, other, ethnographic-based, work in style and social identity construction suggests that persona, or a similarly flexible and dynamic structure, is a more appropriate unit in interactional settings (e.g., Eckert 2008, Bucholtz & Hall 2005). Personae are holistic, embodied identities that often correspond to macro-demographic groups, but are not equivalent. In her 2019 study, D'Onofrio used a cloze test, in which listeners fill in missing words from a passage, with three different visual primes and two different voices. Two of the photos were of the same Korean man, an actor, in two different settings—wearing either a rather North American-looking flannel shirt, or formal wear associated with Korean popular music, known as K-Pop—while the third was of a white man. One of the voices was L1 American English, and the other L2 English with Korean L1. The two different photos of the actor evoke two different types of ethnically Korean people listeners may be familiar with: a Korean or a Korean American. Listeners were most accurate at transcribing the L1 English voice overall, but for the Korean-accented English, the K-Pop persona was linked with more successful transcription than either of the other photos. The Korean American persona patterned with the white American persona, not with the K-Pop performer, despite being the same individual.

In her (2015) paper, which we'll discuss more in Chapter 5, she found that expecting to hear a "Valley Girl" was associated with variants affected by the California Vowel Shift more than a non-specific Californian. Listeners aren't swayed by knowing that someone is a Californian as much as they are by knowing what type of Californian that person is. D'Onofrio's two studies suggest that by considering ethnicity or region alone, researchers might be aiming "too broadly" when it comes to social cues. Support for the concept of persona in sociophonetics is mirrored in production with, for example, Podesva's (2011) work on style shifting performed by a single speaker who inhabits multiple personae in different social settings.

Munson (2010) proposes a mechanism in which phonological knowledge is multi-level. Abstract knowledge emerges from exemplars or distributions. Generalizations about phonotactics, e.g., are based on the lexicon. Developing as a speaker of a language entails gradually more abstract generalizations forming over time. Based on this, Munson proposes a lexicon of talkers. Language users generalize from a lexicon of talkers to speaker attributes. Munson doesn't discuss these attributes in detail, but suggests that they might be indexical fields (Eckert 2008), webs of related, but distinct social meanings, that take on different semiotics depending on the context and the observer. By starting from the talker who commands those indexical meanings, listeners compute whatever stylistic resources they need for production or perception.

This type of approach seems to me to be reflected in the use of voices inherently rich with social cues without honing in on any particular acoustic or social characteristic as a target. The implicit belief here—which I personally find to be a quite reasonable one—seems to be that speakers will sort out the variation and the cues they have experience with, regardless of whether or not the experimenter is aware of it. In fact, this exact process could produce confounds in matched guise tasks, if the experimenter is unaware of variation that is informative below the level of consciousness, such as higher formants. For example, E. King and Sumner (2015) intentionally chose two speakers who were very different across multiple levels of variation: an older Black man and a younger white woman. Responding to these different constellations of features, listeners associate the same semantic prime with different responses. For example, when presented with the prime "yeast," most listeners thought of baking when it came to the older man and yeast infections when it came to the younger woman. Kapnoula and Samuel (2019) treat the entire voice as "indexical information" in their analysis, varied by the speaker's gender expression. While these approaches make a great deal of sense, they would ideally be complemented by careful ethnographic and experimental investigations into the specifics of the variation. With this information available, the lessons learned can be generalized more easily.

Social identities can also be conveyed through other phonetic features of the speech. For example, Strand (2000) finds that listeners respond faster to voices that are stereotypical (as judged by a separate sample of listeners) for the gender categories "male" and "female," and that perception of sociolinguistic cues related to gender also varies according to the gender-stereotypicality of the voice. In an exemplar model with resonance, as Johnson (2006) proposes, inspired by Goldfinger's (1998) "echoes," evidence for category membership accumulates over the course of exemplar activation. Information gleaned from speech can re-enter the categorization "loop" and increase the activation level of consistent exemplars (see Figure 2.5). I have found a similar effect in my work wherein listeners seem to use formant information within a whisper to make categorical inferences about gender that are reflected in their model of the modal voice. All voices contain information that links to social identities, and there is compelling evidence that listeners, inside and outside of the sound booth, are adept at extracting that information and using it to their advantage in perception.

Above, we explored some of the different ways that social cues and social information have been treated in experimental work. The inconsistencies with how social identities are treated, and the consequences for both producers and consumers of research, became especially salient to me while teaching an undergraduate course, Language and Gender. For example, some bisexual students told me they were uncomfortable with seeing “themselves” lumped in with either gay and lesbian participants, because they aren’t straight, or with straight participants, because they aren’t gay or lesbian. Other sexual identities, such as asexuality, demisexuality, or pansexuality, are often not mentioned at all. Queer identities are carved up, adequated, and erased in experiments and in the literature in ways that reproduce the exclusion these speakers experience both from straight people and within Queer communities. In my role as instructor, I explained the trade-off we must make as researchers between generalizability and specificity, between scientific control and ecological validity. What makes an analysis possible for a researcher does not necessarily reflect their views on the world, and no single investigation could ever tell the whole story.

But I want to call attention to this tension here. I believe that our work and our relationships with the speakers whose linguistic knowledge we benefit from will be strengthened by not ignoring it. I am also reminded of Sumner’s presentation during the Sociolinguistic Cognition symposium I co-organized at the 2018 Linguistic Society of America meeting, in which she cautioned about extrapolating from an individual to a group effect in speech perception. That is, she problematized a very common approach, one which I myself have adopted in experimental work, one which sometimes seems impossible to avoid, in which a single speaker represents a whole group of speakers. Some of the features of the stimulus person’s speech will in fact be attributable to the target axis of variation. However, other features may be idiosyncratic, or representative of the intersection of the targeted identity with the speaker’s other identities. As Sumner noted, there isn’t an easy way around this, aside from using different speakers with overlapping identities in follow-up experiments to try to tease out specific cues.

I want to acknowledge that many of us would likely take issue with the adequation inflicted on both listeners and speakers in phonetics experiments if they occurred in other contexts. Intersecting or very marginalized identities are often ignored, while single speakers bear the burden of representation for entire heterogeneous groups. I am not yet sure how we can do better, but I believe that we must try, both to bolster the ecological validity of our research and to produce work that we are proud to show our students who experience exclusion in their everyday lives.

Chapter 4:
The ExemPy library

This chapter introduces ExemPy, the library I've written in Python for modeling spoken language perception using the Generalized Context Model. Before explaining more details about ExemPy and how to use it, let's first consider how simulations can contribute to phonetics research. Once that's established, we'll move on to some specifics of the library and how to use it. Finally, we'll see some examples of how ExemPy can be used in research contexts. This chapter overlaps with a manuscript currently under review which was co-authored with Sarah Bakst, but represents my own work. Corresponding code is provided in the appendix and on GitHub. The library and documentation are here: https://github.com/emilyremirez/ExemPy/tree/main/ExemPy; the demonstrations are here: https://github.com/emilyremirez/ExemPy/tree/main/Dissertation%20demos

4.1 Motivations and Considerations for Simulating Speech Perception

ExemPy's central use case is to categorize a set of stimuli based on a provided set of "exemplars." The resulting dataframe resembles the aggregated data of a perception experiment. Each stimulus is assigned a "response" and a probability for that response. For the purpose of comparing simulated and behavioral data, we take this probability as a stand-in for the proportion of responses.

I find it worth stating explicitly that I take this model as a descriptively useful abstraction, not a literal account of cognitive processes. A simulation of speech perception is not a substitute for behavioral evidence, but it can be used as a tool in combination with descriptive and behavioral work. I want to call particular attention to the potential of simulated experiments to make predictions and explore theoretical spaces.

With simulations, researchers have precise control over each parameter that's been hypothesized. This lets us visualize what behavior would look like if different accounts were true. We can compare these predictions to existing evidence or use them to motivate new investigations. In incorporating simulations like ExemPy into experimental work, it's important to be clear and specific about what the simulation can and can't do. Most importantly, we can't expect this sort of illustration to tell us something new about the exemplar dataframe we use. Rather, we should conceptualize this type of research as a concrete answer to the question, “what if this account were true?”

There are also cases where a perception result would be useful, but isn’t possible. For example, when designing experiments, it can be helpful to know things like how similar two sounds are for listeners. If that research doesn’t yet exist, a simulation can provide some basis for the design. A sleeping language has no living first-language users, but a well-motivated simulation can approximate a behavioral result that isn’t possible to observe. Learning to use the library and running the code require significantly less investment, creating new opportunities for students, for example, who may have limited funding or training to explore their data.

As with any methodology, there are limitations we must continually address while using ExemPy. Implementing a model forces us to make choices that we may be theoretically agnostic to. For example, the code may require two processes to happen sequentially, and so one of those steps has to happen first. A theory may tell us that some value increases, but we have to decide whether to add or multiply. Every parameter requires some value, and it isn’t always obvious what that should be. When we choose a dataframe to act as a set of exemplars, we must artificially constrain their properties relative to the rich, embodied, multimodal experience of using language. We choose an artificial number of dimensions that we think may be important, and we choose how to measure or record them. We also choose how many observations of each category we include. Further, participants in behavioral tasks have wholly individual experiences with language, while the ExemPy “perceiver” has only one set of experiences. I accept these limitations because there’s value in what they make possible. Every individual decision should be considered with skepticism. As a whole, they work together to give us an impression that’s “close enough.” That is, useful, if imprecise.

A further aspiration is to contribute infrastructure for bridging between subfields. This type of collaboration and cross-pollination enriches both the research and the researcher. We can’t make robust, generalizable claims about language using data only from overrepresented colonial languages. The easier it is for people with the relevant data to participate in modeling, the more perception researchers can benefit from their work. Language documentarians who collect phonetic data from spoken language production can leverage simulations to offer insight into both that language in particular and perception in general.

In writing the library, I drew on my own experiences incorporating computation into research and watching others do the same. I don't think I'm alone in saying that I developed my programming skills primarily by editing snippets of code I got from other people. Further, the lessons really don't stick unless I'm invested in the application. A motivating ambition was to provide a friendly place to get that starting code. I tried to anticipate what users would want to do and what questions or mistakes might come up along the way. I worked with friends' data to generalize the library and get their feedback on usability. My muse was a hypothetical researcher who wants to develop computational skills and has a question about language that perceptual simulation could inform. By working through the routines in the documentation and editing the code to their needs, they could contextualize programming concepts and build confidence. They don't need to have secured funding, for example, to begin exploring their data.

4.2 Overview of ExemPy

This section provides a narrative overview of the ExemPy library, from the perspective of common workflows. Detailed documentation of the code is available here: https://github.com/emilyremirez/ExemPy/tree/main/ExemPy A notebook overview is available here: https://github.com/emilyremirez/ExemPy/blob/main/Dissertation%20demos/Basics.ipynb. All code is also included in the appendix.

Types of simulations

So, if our goal is to compare the results of the model with observable behavior, we begin with reproducing experimental conditions. Currently, ExemPy simulates two types of tasks:

1. Identification tasks, in which the "perceiver" chooses the most probable label, from among all those represented in the exemplar cloud
2. Forced choice tasks, in which the "perceiver" chooses which of two options is more likely

Both of these tasks are simulated using the wrapper function `multicat()`, which categorizes multiple stimuli. As a wrapper function, it calls a series of functions corresponding to smaller steps. Equations 2.1-2.3 of the Generalized

Context Model (see Chapter 2) are implemented as part of the `activation()` function; `probs()` implements Equations 2.4 and 2.5; and `choose()` selects the category label with the highest probability as the “percept.”

During a forced choice task, an experiment participant must choose between a limited set of options. Often, these choices are two clear endpoints of an interpolated continuum. But what does it mean for a perceiver to know that they'll have to pick between two choices? At this time, I identify three, non mutually exclusive vectors for modeling the task effects on perception. The first two are implemented through during the `activation()` function, adjusting the weights of individual exemplars (*N*) or relevant features in the contrast (*w*).

The expectation within the experiment is that the percept will be one of few, delimited options. Base activation *N* may be raised for exemplars belonging to the category of the alternatives and dampened for would-be competitors in a straightforward example of priming. This may also affect the way a listener attends to features. Increasing the weights for the dimensions the two alternatives tend to differ along would maximize the psychological distance between the categories and minimize the differences within categories.

The third place to model the effects of the experiment design is during the `choose()` function. Forced choice can be implemented after probability has been calculated. Rather than choosing the category label with the highest probability overall, the “perceiver” chooses the alternative with the higher probability of the two.

As much as possible, I tried to keep this customizable by the user, but I do model a default behavior. In the current implementation, probabilities are calculated as usual, but the `choose()` function chooses between two alternatives.

Setting attention weights

We have hypotheses about the role attention weights serve theoretically, but the model requires us to set actual values. I used `scipy.optimize.minimize()` to generate values for *w* that result in “accurate” categorization.

Because there are so many variables, there are numerous possible solutions that result in low error. Different sets of local minima could be found depending on the starting values of the weights. To address this, I run `minimize()` multiple times with multiple, random initial guesses. Anchoring the first parameter at 1 and using guesses between 1 and 3 constrains the solution space. This greatly reduces the duration of the optimization procedure.

Once the process is finished, the researcher may need to choose a single set of values for the simulations. This is ultimately a judgment call, but I consider three factors. First, I choose a *w* with a low error rate. However, error rates across trials tend to be consistent, regardless of the specific values fit. In deciding between low-error solutions, I take a "common sense" approach of prioritizing weights that, second, match behavioral evidence and, third, are similar orders of magnitude. The alternative, which I adopt below, is to use multiple generated sets of values.

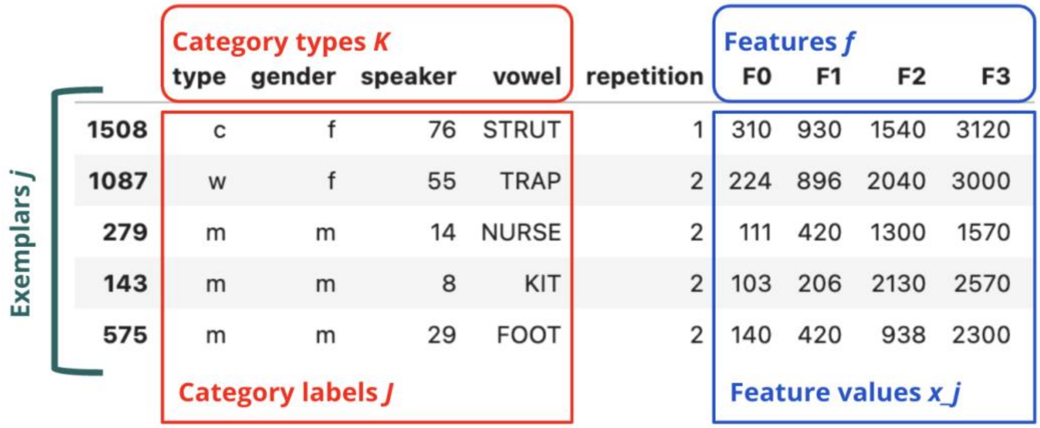
The exemplar cloud

The "exemplar cloud," or set of stored memories, is represented by a Pandas dataframe, which can be created from a spreadsheet or table. Each row represents an exemplar or observation.

There are two types of columns: categories and dimensions. Figure 4.1 shows a sample of the dataframe I used to develop this library, from Peterson and Barney's famous 1952 study. In this example, the category types are speaker type (defined below), speaker gender, speaker id, and vowel. The dimensions are fundamental frequency (F0) and the first three formant frequencies (F1-F3). You'll notice that Figure 4.1 also includes the variable repetition. For this work, we aren't interested in the difference between repetitions 1 and 2; however, including it in the dataframe allows to locate individual observations within the code.

The stimulus, *i*, or set of stimuli, should also be a dataframe with at least one row. In the exemplar cloud, every cell must be filled. Each exemplar has a category label for each category type, and a value for each feature. For the stimuli being categorized, only the dimension values are necessary. It's crucial that the columns are named identically, and that these same names are used in the dictionary specifying attention weights for each dimension. In most of our examples, we draw the stimuli from the same dataframe as the exemplar cloud.

Choosing an exemplar cloud necessitates some artificial assumptions. First, the researcher chooses what the dimensions are and represents them as static points in time. In experimental research, stimuli have often been designed to emphasize and isolate particular cues. Because of this, we attribute differences in behavior to those differences in the stimuli. And, the embodied experience of perceiving contains an unknowable number of cues, including information about how they change over time. Johnson (2006) addresses this by using auditory spectrograms, representations of the physical properties of sounds and the physiology of hearing. Even this removes the context and detail fundamental to the theory. The values chosen also tend to be continuous and numeric. Second, it delineates a limited number of exemplars, likely not scaled for frequency.


Category types K
Features f
Exemplars j
Category labels J
Feature values x_j

	type	gender	speaker	vowel	repetition	F0	F1	F2	F3
1508	c	f	76	STRUT	1	310	930	1540	3120
1087	w	f	55	TRAP	2	224	896	2040	3000
279	m	m	14	NURSE	2	111	420	1300	1570
143	m	m	8	KIT	2	103	206	2130	2570
575	m	m	29	FOOT	2	140	420	938	2300

Figure 4.1: A sample of an exemplar cloud dataframe. Data from Peterson & Barney 1952, via Barreda 2015.

The dataframe or spreadsheet format is an abstraction, but it's a useful one. The format is familiar and will allow many researchers to "plug and play" using existing datasets. Its limited scope is also an advantage when it comes to controlling for specific factors.

4.3 Demonstrations with Peterson & Barney

The data in these demonstrations are from Peterson & Barney 1952. One reason I chose this dataset is that other phoneticians will be very familiar with it as a landmark study of US English vowels. It's been used in other simulations (e.g., Ames & Grossberg 2008), and using an established set of exemplars lets us compare results more directly. Further, we know how at least one set of human listeners respond to speech synthesized from these formant values (Hillenbrand & Gayvert 1993).

The dataset is tagged for both linguistic and social categories. 76 speakers repeated 10 vowels twice each. I represent these vowels using lexical set notation; each word stands for the vowel it contains. The speakers are divided into 3 “types”: man (33 speakers), woman (28), and child (15). While this reflects a limited and binary view of gender, it allows us to consider how these types of categories interact.

I downloaded the dataset from the phonTools R package (Barreda 2015). Prior to working with the data, I changed the X-SAMPA vowel transcriptions to lexical set notation (Wells 1982), to make it easier to code with and interpret.

Lexical Set	**IPA**	**X-SAMPA**	**Arpabet**
FLEECE	i	i	IY
KIT	ɪ	I	IH
DRESS	ɛ	E	EH
TRAP	æ	}	AE
STRUT	ʌ	V	UH
PALM	ɑ	A	AH
THOUGHT	ɔ	O	AW
FOOT	ʊ	U	OO
GOOSE	u	u	UW
NURSE	ɝ	3	ER

Table 4.1: Lexical set, IPA, X-SAMPA, and arpabet notation of vowels included in dataset

Identification task

To simulate an identification task, I categorized the entire Peterson and Barney (1952) dataset. Each row in the data frame is compared to every other row, except for itself. Researchers using ExemPy can choose other exemplars to exclude during activation. For example, perceiving a novel talker can be simulated by not comparing stimuli to exemplars from the same speaker.

Doing this allows us to compare our results to human behavior. A model is judged not just on how it gets things right, but how it gets things *wrong*. That is, when human listeners responded to these stimuli, they sometimes mistook one vowel for another. Informative patterns in these mistakes are captured in confusion matrices. These tables show how often a stimulus was given a particular response. If perceivers regularly confuse one sound for another, that tells us those sounds are perceptually similar.

ExemPy's `confusion()` function will generate a dataframe like the one in Figure 4 below from a dataframe of choices. The columns here represent the stimulus, while the rows represent the response. The diagonal captures accurate identification, and has the highest values. That is, in all cases, the "listener" was more likely to be right than wrong.

The results of the simulation are highly correlated with the experimental results reported by Hillenbrand and Gayvert (1993). Again, the exact results will vary depending on the attention weighting parameters, but the results of this example are representative of other simulations, as demonstrated in Table 4.2.

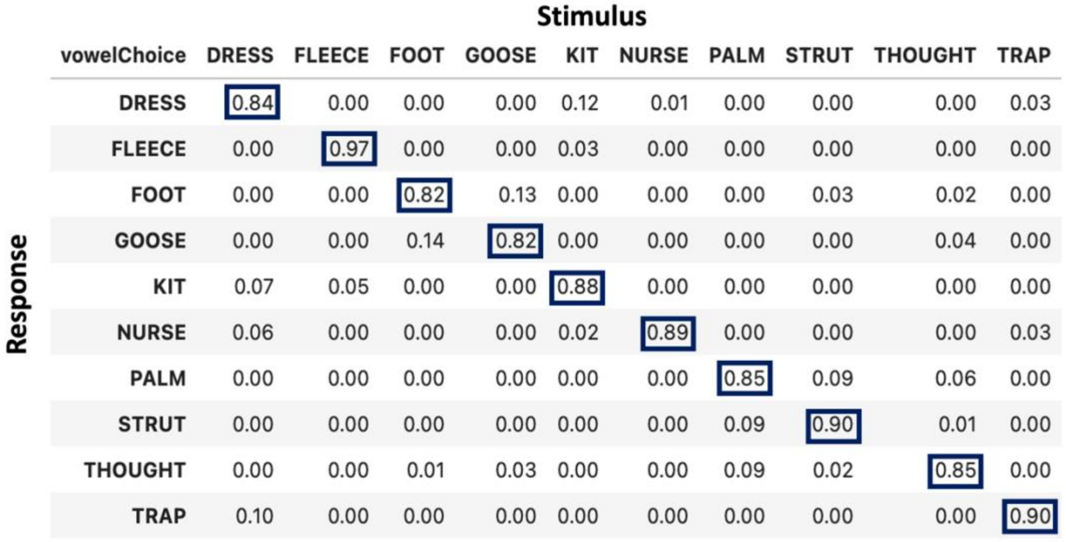
To compare the confusion tables, I use two measures. The root mean square (RMS) distance is a measure of the difference between two matrices, while Pearson's correlation (*r*) measures their similarity. If the tables are similar, RMS will be low and *r* will be high. To calculate these results, I first excluded every 0.0 from the table. Because 0 correlates perfectly with 0, skipping this step would artificially inflate the correlations.

To create Table 4.2, I fit attention weights and simulated the identification task 5 times. For each set of parameters, z0 (fundamental frequency in Bark) was anchored at 1. That is, the other weights will be set assuming that z0 will be 1, accelerating the parameter fitting process. Note that the attention weights shown in the table have been normalized to sum to 1, rounded to 2 decimal places. For all 5 trials, RMS Is between 0.073 and 0.075, and *r* is between 0.964 and 0.966. The minimized error rate is between 0.109 and 0.118.

Forced choice task

https://github.com/emilyremirez/ExemPy/blob/main/Dissertation%20demos/Forced%20choice.ipynb

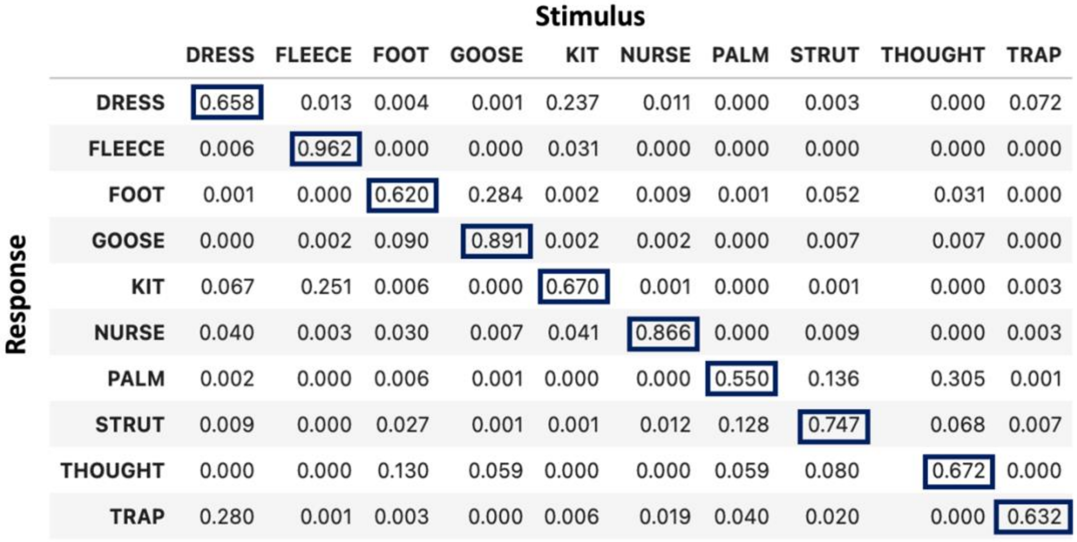
To simulate a forced choice task, I first created three sets of "stimuli" using ExemPy's `continuum()` function. This function creates interpolated continua between two end points. These 7-step continua started with average values for FOOT on one end, moving in even increments towards the value for STRUT on the other. I did this using average values for the three speaker types: w[oman], c[hild], or m[an].



Stimulus

	vowelChoice	DRESS	FLEECE	FOOT	GOOSE	KIT	NURSE	PALM	STRUT	THOUGHT	TRAP
Response	DRESS	0.84	0.00	0.00	0.00	0.12	0.01	0.00	0.00	0.00	0.03
	FLEECE	0.00	0.97	0.00	0.00	0.03	0.00	0.00	0.00	0.00	0.00
	FOOT	0.00	0.00	0.82	0.13	0.00	0.00	0.00	0.03	0.02	0.00
	GOOSE	0.00	0.00	0.14	0.82	0.00	0.00	0.00	0.00	0.04	0.00
	KIT	0.07	0.05	0.00	0.00	0.88	0.00	0.00	0.00	0.00	0.00
	NURSE	0.06	0.00	0.00	0.00	0.02	0.89	0.00	0.00	0.00	0.03
	PALM	0.00	0.00	0.00	0.00	0.00	0.00	0.85	0.09	0.06	0.00
	STRUT	0.00	0.00	0.00	0.00	0.00	0.00	0.09	0.90	0.01	0.00
	THOUGHT	0.00	0.00	0.01	0.03	0.00	0.00	0.09	0.02	0.85	0.00
	TRAP	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.90

a) GCM



Stimulus

		DRESS	FLEECE	FOOT	GOOSE	KIT	NURSE	PALM	STRUT	THOUGHT	TRAP
Response	DRESS	0.658	0.013	0.004	0.001	0.237	0.011	0.000	0.003	0.000	0.072
	FLEECE	0.006	0.962	0.000	0.000	0.031	0.000	0.000	0.000	0.000	0.000
	FOOT	0.001	0.000	0.620	0.284	0.002	0.009	0.001	0.052	0.031	0.000
	GOOSE	0.000	0.002	0.090	0.891	0.002	0.002	0.000	0.007	0.007	0.000
	KIT	0.067	0.251	0.006	0.000	0.670	0.001	0.000	0.001	0.000	0.003
	NURSE	0.040	0.003	0.030	0.007	0.041	0.866	0.000	0.009	0.000	0.003
	PALM	0.002	0.000	0.006	0.001	0.000	0.000	0.550	0.136	0.305	0.001
	STRUT	0.009	0.000	0.027	0.001	0.001	0.012	0.128	0.747	0.068	0.007
	THOUGHT	0.000	0.000	0.130	0.059	0.000	0.000	0.059	0.080	0.672	0.000
	TRAP	0.280	0.001	0.003	0.000	0.006	0.019	0.040	0.020	0.000	0.632

b) Hillenbrand & Gayvert 1993

Figure 4.2: Confusion matrices from (a) one round of identification task and (b) Hillenbrand & Gayvert.

Figure 4.3 was created using the `cpplot()` (categorical perception plot) function in ExemPy. It shows the results of categorizing these three sets of continua. The X-axis shows the step along the continuum, with 1 being distinctly FOOT, 7 being distinctly STRUT, and the rest being somewhere in between. The Y-axis represents the probability of a response, taken as the proportion of people who would have given that response in a listening experiment. The gray dotted line at 0.5 visualizes chance. The point at which the curve intercepts the 0.5 line is considered the “boundary” between those two categories. This graph shows that our “listener” draws the boundary between FOOT and STRUT differently for the type m stimuli than for w and c.

Remember that ultimately the more similar an exemplar is to the stimulus, the more likely it is that the stimulus will be categorized the same way as the exemplar. Based on past experience, listeners know that the difference between FOOT and STRUT depends in part on the overall frequencies and whether those qualities are more associated with men or with women and children.

In Figure 4.3, there is no point for step 4 in the woman stimuli. This is because the most probable response was neither FOOT nor STRUT, but PALM. To plot any point along this axis would imply an artificially high probability of one of the two options. When running the routine in ExemPy, the user will get a printout notifying them, along with the step number, category label choice, and probability.

	Attention weight *w* values					**Comparisons**	
Trial	**z0**	**z1**	**z2**	**z3**	**error**	**RMS**	**r**
1	0.10	0.37	0.23	0.29	0.109	0.073	0.964
2	0.17	0.46	0.22	0.15	0.118	0.075	0.964
3	0.27	0.43	0.16	0.15	0.109	0.074	0.966
4	0.14	0.35	0.26	0.25	0.109	0.073	0.965
5	0.18	0.31	0.29	0.21	0.115	0.075	0.966

Table 4.2: Comparisons of confusion matrices categorized using 5 different sets of attention weights

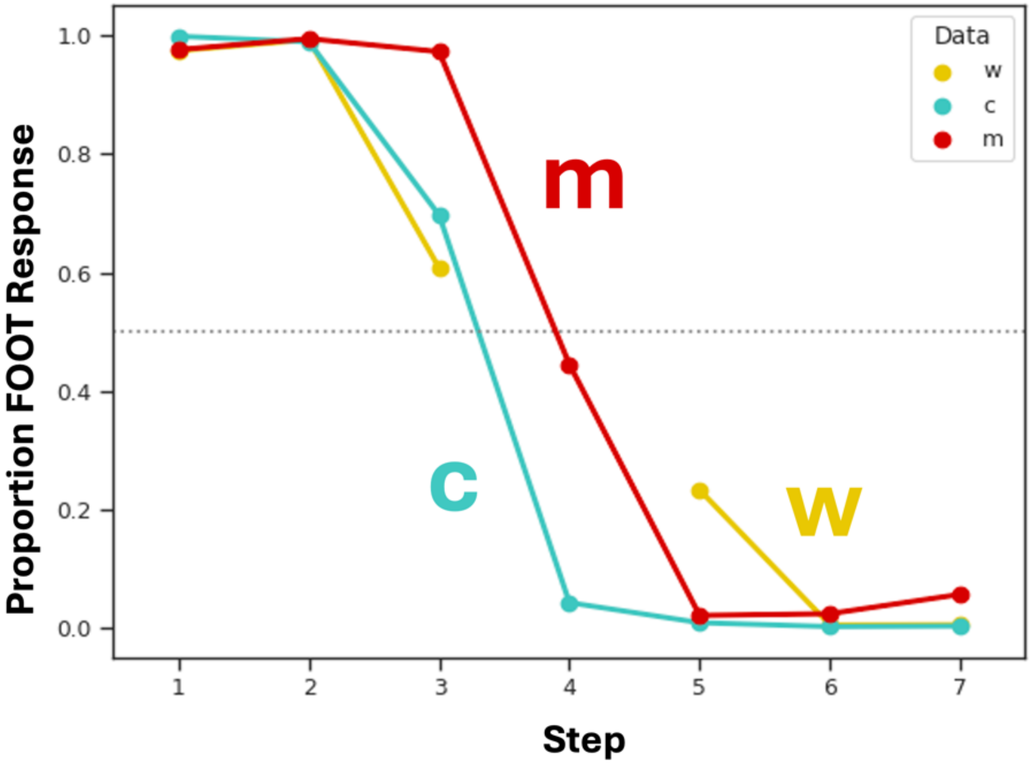


Figure 4.3: Results of simulated forced choice task between FOOT and STRUT

4.4 Using ExemPy to understand altered auditory feedback: Co-authored work with Dr. Sarah Bakst

In a collaboration with Dr. Sarah Bakst, we used ExemPy to simulate real-time data from an experiment she had conducted using altered auditory feedback to investigate vowels in Tamil. After working together to identify a relevant question, I provided some starter code for Sarah to use in her analysis. From there, she conducted the simulations and interpreted the results, with some feedback from me. Through this collaboration, we were able to analyze her existing data in a novel light.

This process was also crucial for making ExemPy more generalizable and user-friendly, functioning almost like a beta test. The insightful feedback from a trusted collaborator identified targets for more robust documentation or exception handling, for example. Ensuring the code worked with Dr. Bakst's data forced technical debt to be repaid and hard-coding shortcuts be undone. I am very grateful to have had such an intelligent, empathetic, and pleasant partner for this task.

As Dr. Bakst writes in our manuscript, “In an altered auditory feedback experiment, participants hear a version of their speech that does not match what they actually produced. For example, a formant may be altered in near-real time so that participants hear themselves producing a different vowel than they intended” (Remirez & Bakst 2023). As speakers adjust their articulation in response to these perturbations, the vowel they produce could have been perceived as the same category, or a different category. In other words, their articulation may or may not cross a would-be category boundary. The relevant question is perceptual, not acoustic, but we only have access to these speakers’ productions, as well as what they would have heard themselves say. Using ExemPy to simulate the categorization can fill this gap. The model identifies altered utterances that were likely to have crossed the perceptual boundary. In the absence of empirical data, the simulation is leveraged to make another analysis of the acoustic data possible.

Using this methodology, she finds that the GCM more or less replicates a lab finding in which category boundaries were derived empirically:

Niziolek et al. (2013) conducted an altered feedback study where category boundaries were empirically derived. These category boundaries were then used to determine which trials in the altered feedback study would have caused participants’ feedback to cross category boundary lines–that is, where speakers would have heard themselves producing a categorically different vowel from the target. On boundary-crossing trials, speakers showed a greater degree of perturbation-opposing behavior. In the experiment presented here, speakers who experienced a higher percentage of category-crossing shifts also showed more feedback-opposing behavior on average, but not on all the individual trials where these category-crossings occurred. That said, out of all the trials, those with the greatest amount of perturbation opposition occurred when the heard signal crossed simulated category boundary lines (Remirez & Bakst 2023).

By comparing the results of the simulation to lab-based insights, we lend credibility to the methodology. This showcases ExemPy as a viable, useful resource for understanding empirical data.

4.5 Using ExemPy to model priming and resonance

As previewed in the introduction, ExemPy can be used to explore a theoretical space. Here I explore two facets discussed in sociophonetic priming, activation raising (Chapters 2 and 3) and resonance (Chapter 2). I model the effect of each process on activation.

In order to conduct the simulations, I first identified an ambiguous stimulus: Speaker 47's second repetition of the FOOT vowel. I chose this stimulus because the probability of both speaker type and vowel was consistently low. Classifying both repetitions using the weights given in Table 4.2, yields 10 sets of vowel and speaker type probabilities. Of these 10 categorizations, the speaker was mis-categorized as a child 3 times. Vowel choice–FOOT–was always accurate, but probabilities were low (details in Table 4.4 below).

Phonetic details of that stimulus, along with two points of comparison, are given in Table 4.3. Comparing repetition 2 to both the speaker's other repetition and mean values for this vowel across type w speakers, we see in particular that the fundamental frequency, 205 Hz, is lower than both the other repetition (286 Hz) and the type w average (234 Hz). This is especially interesting given that fundamental frequency is closely linked to the percept of pitch, which is in turn linked to gender performance and ideology. In other words, this observation may be comparable to the type of non-stereotypicality manipulated in Strand's experiments (Strand 1999, Strand & Johnson 2000, Johnson 2006).

	F0 (Hz)	**F1 (Hz)**	**F2 (Hz)**	**F3 (Hz)**
Ambiguous stimulus: Speaker 47, rep 2	205	570	1200	2970
Speaker 47, rep 1	286	540	1200	2860
Average w values	234	469	1161	2685

Table 4.3: Phonetic details of FOOT productions.

Specifically, probabilities are split across four other vowels, as shown in Table 4.4. As expected, the exact ratio depends on the attention weighting, but the two biggest competitors are STRUT and THOUGHT, followed by GOOSE and PALM.

w	FOOT	STRUT	THOUGHT	GOOSE	PALM
1	0.419	0.134	0.390	0.033	0.023
2	0.271	0.266	0.264	0.058	0.141
3	0.368	0.340	0.196	0.041	0.054
4	0.369	0.270	0.226	0.053	0.082
5	0.440	0.370	0.073	0.044	0.073
mean	0.373	0.276	0.229	0.046	0.037

Table 4.4: Probabilities of classification as different vowels, rounded to three decimal places, using the attention weights in Table 4.2.

In order to consider the effects of resonance and category priming, we'll be looking at plots like the one in Figure 4.4 below. The graph plots each exemplar in the cloud in F1, F2 space (Bark). As customary, axes are inverted to correspond more closely to articulatory space. Vowel categories are separated by color. The size of the exemplar corresponds to how activated it is by the stimulus. The stimulus is not plotted as an exemplar in the graph because it was excluded from comparison. That is, the stimulus was not compared to itself. For reference, the stimulus is marked instead with a dark blue x.

For these figures, I've chosen to use the 2nd set of exemplar weights, where FOOT is the winner, but competition is tight with STRUT and THOUGHT. Top-down effects are especially observable where uncertainty is high, so starting from a place of high competition among vowels is more likely to demonstrate the effect we're after.

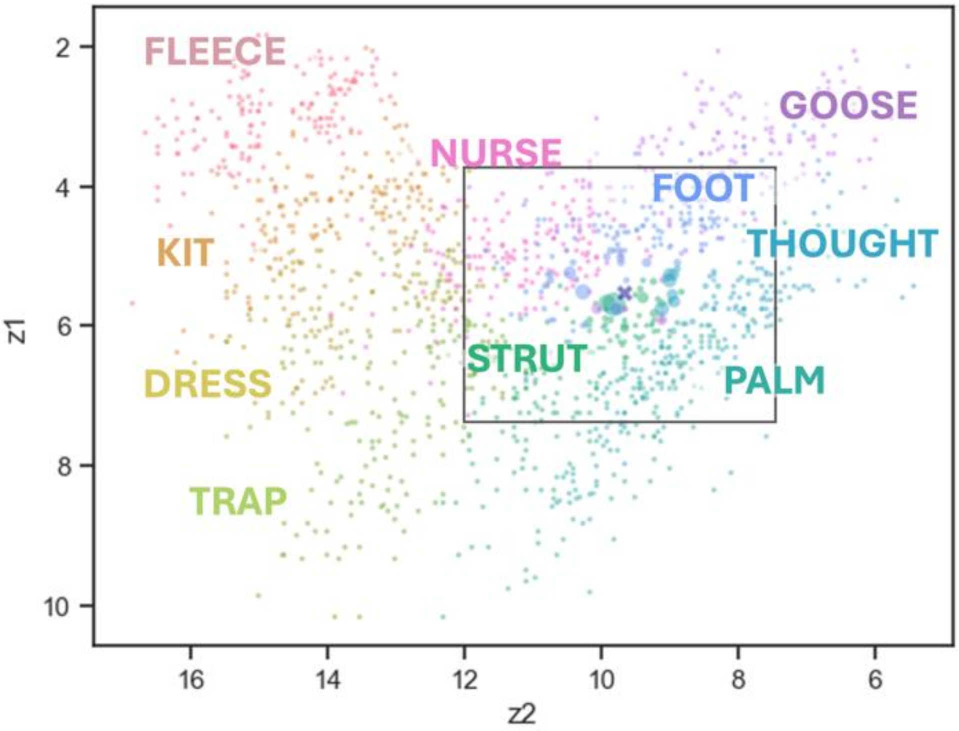


Figure 4.4: Activation of exemplars by vowel, in acoustic space. Color marks vowel category, and size marks amount of activation. The stimulus is represented by the dark blue x.

In the rest of this section, we'll compare these activations under certain conditions. First, recall that the ambiguous stimulus–Speaker 47, FOOT, repetition 2–has an unusually low F0 compared to the other repetition (286 Hz), the overall type w average F0 for FOOT (234 Hz), and the speaker's average F0 across all repetitions of all vowels (253 Hz). (The lowest F0 for a type w speaker in the Peterson and Barney data is 150 Hz.) F0 is linked to perception of gender, so "inducing expectation" that the speaker is a woman by raising the resting activation scaling for women speakers will push the vowel more towards veridical perception. The amount of scaling, of course, determines the effect. In Table 4.5, we see the effects of setting resting activation for type "woman" exemplars to 2, 5, 10, and 25. That is, where "man" and "child" had a resting activation of 1, activation for "woman" exemplars was multiplied by these values. (Recall Equation 2.3, which defines activation of an exemplar *a_ij* as similarity scaled by resting activation *N*.) Likewise, we can predict the output of expecting a "man" or "child", making the vowels STRUT and THOUGHT, respectively, more likely. That is, the vowel percept changes depending on speaker type. The effect on activation for 10 times bias towards each type is shown in Figure 4.5. In each case, we can see that the most activated exemplars are those close to the stimulus–those that really are similar. However, the category that the most activated exemplars belong to depends on the "expectation." (The category whose members have the highest activation, of course, corresponds to the highest probability.)

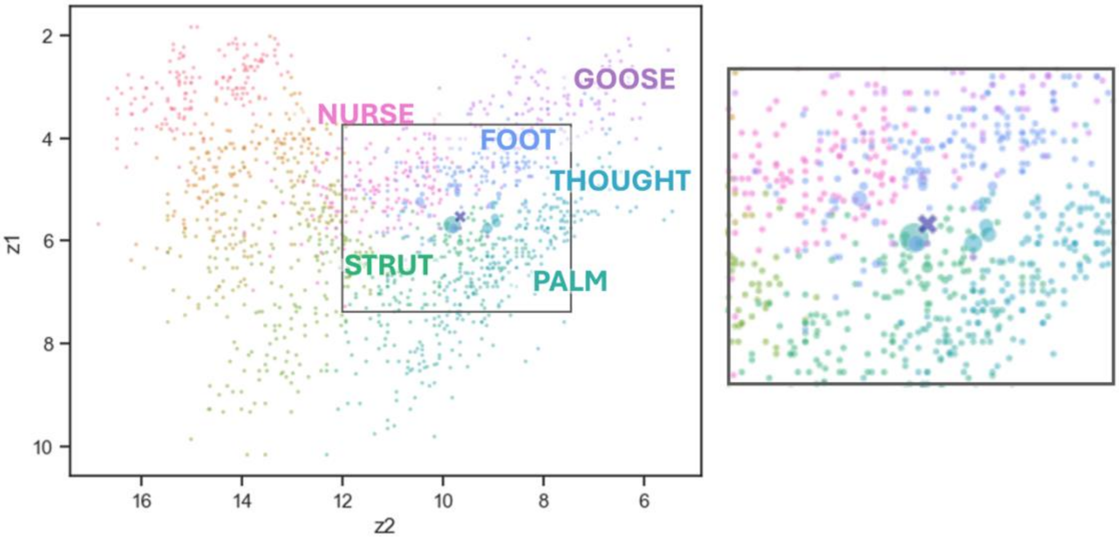
N	Category	FOOT	STRUT	THOUGHT	GOOSE	PALM
w: 1	FOOT	0.271	0.266	0.264	0.058	0.141
w: 2	FOOT	0.293	0.199	0.274	0.040	0.193
w: 5	FOOT	0.317	0.127	0.285	0.021	0.249
w: 10	FOOT	0.328	0.093	0.290	0.012	0.176
w: 25	FOOT	0.336	0.069	0.294	0.005	0.296
m: 10	STRUT	0.088	0.751	0.093	0.180	0.049
c: 10	THOUGHT	0.355	0.072	0.379	0.156	0.038

Table 4.5: Probability of categorization of each vowel label with different activation scaling for types. If the value is not given, the resting activation was set at 1.

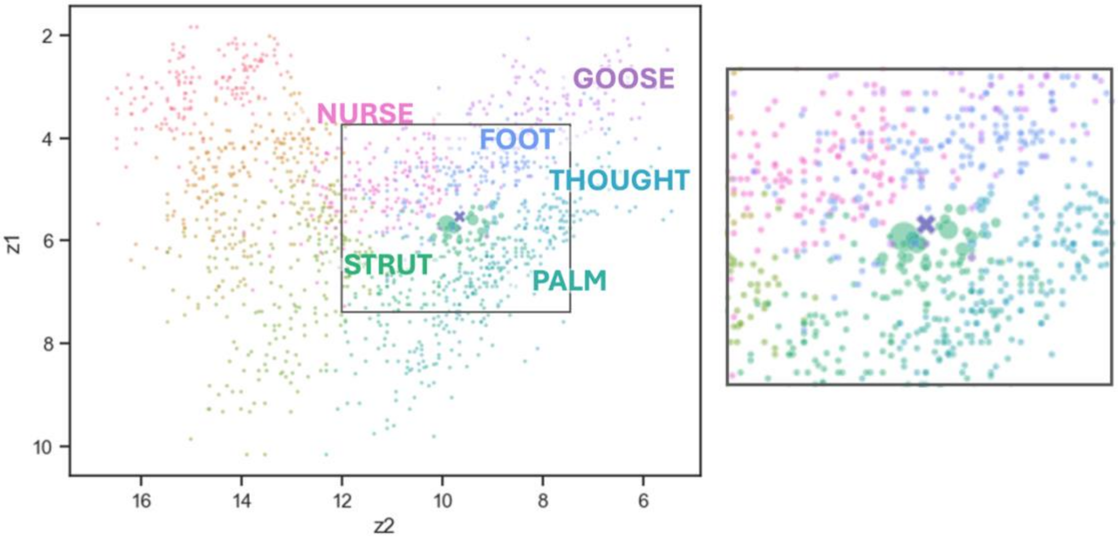
The effect is the least noticeable in Figure 4.5a, where woman exemplars are more activated. Activation increases for several similar FOOT exemplars and a single STRUT exemplar very close to the stimulus in F1/F2 space. We may expect the activation of STRUT to correspond to an increase in its probability; however, the single exemplar is not enough to have an observable effect. When a man speaker is "expected," however, activation shifts to several STRUT exemplars, enough to change the classification from FOOT to STRUT. Likewise, in 4.5c, activation shifts toward THOUGHT when a child is expected, changing the percept of the vowel. With this priming, exemplars that are less similar are still able to contribute significant activation.

In the prior paragraphs, we manipulated the type "expectation" that occurs "outside" the speech perception event. This is a proxy for the sort of "non-linguistic" priming invoked through images or labels. Now, let's consider the other mechanism we've discussed: resonance. To simulate resonance, we first calculate activation and probability as usual. Resting activation is then incremented by a “resonance term”: the probability of categorization with the category level is divided by the number of cycles. Taking the probability allows for gradient activation spreading, while dividing by a number of cycles constrains the term from becoming too large. In this implementation, one category is set to be resonated on, and have its probabilities reenter the categorization loop through an increase in activation. For these demonstrations, we’ll use speaker type. To schematize the process being described, the listener would hear the stimulus, make a preliminary decision about the type of speaker, and then use this information to modulate expectations about vowel categories. As seen in Table 4.6, the effect is similar to that of priming, but is generated bottom-up, without external priming. In this case, the change in activation too minimal to be observed in the plots, but its evidence is seen in changes to probabilities.

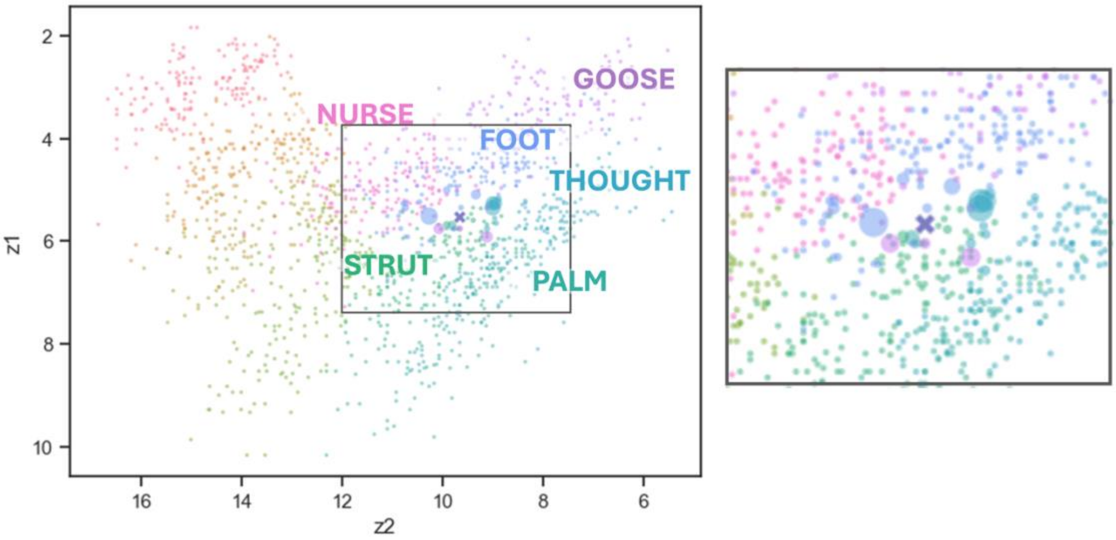
This section demonstrated some of the knobs that can be adjusted using ExemPy to probe the approximate different theoretical assumptions. While it’s worth re-stating the risks of taking this kind of parameter fitting too literally, if what we’re truly interested in is human behavior, the output can be compared to experimental results. The experiment described in the next chapter was designed with this type of simulation in mind. Can we observe a difference in the modeling, and does that difference correspond to what we observe in behavior? A correspondence between the two would provide supporting evidence for that model. Further, this simulation has pedagogical implications for explaining concepts like priming and resonance, which can be opaque. Not only can the results of the simulations be used as illustrations, a learner could easily make their own tweaks and observe the output. This was a specific goal in writing the library: A non-expert should be able to quickly test predictions in order to test and expand their understanding of the theory.


GOOSE
NURSE
FOOT
THOUGHT
STRUT
PALM
z1
z2
2
4
6
8
10
16
14
12
10
8
6

a) w: 10


GOOSE
NURSE
FOOT
THOUGHT
STRUT
PALM
z1
z2
2
4
6
8
10
16
14
12
10
8
6

b) m: 10



c) c: 10

Figure 4.5: Reproduction of Figure 4.4–exemplars in acoustic space, with marker size corresponding to activation level–with priming toward different speaker types.

Cycles	FOOT	STRUT	THOUGHT	GOOSE	PALM
0	0.271	0.266	0.264	0.058	0.141
1	0.277	0.249	0.268	0.056	0.150
5	0.283	0.232	0.271	0.053	0.160
10	0.285	0.225	0.273	0.052	0.164

Table 4.6: Probability of categorization of each vowel label with different numbers of resonance cycles on speaker type.

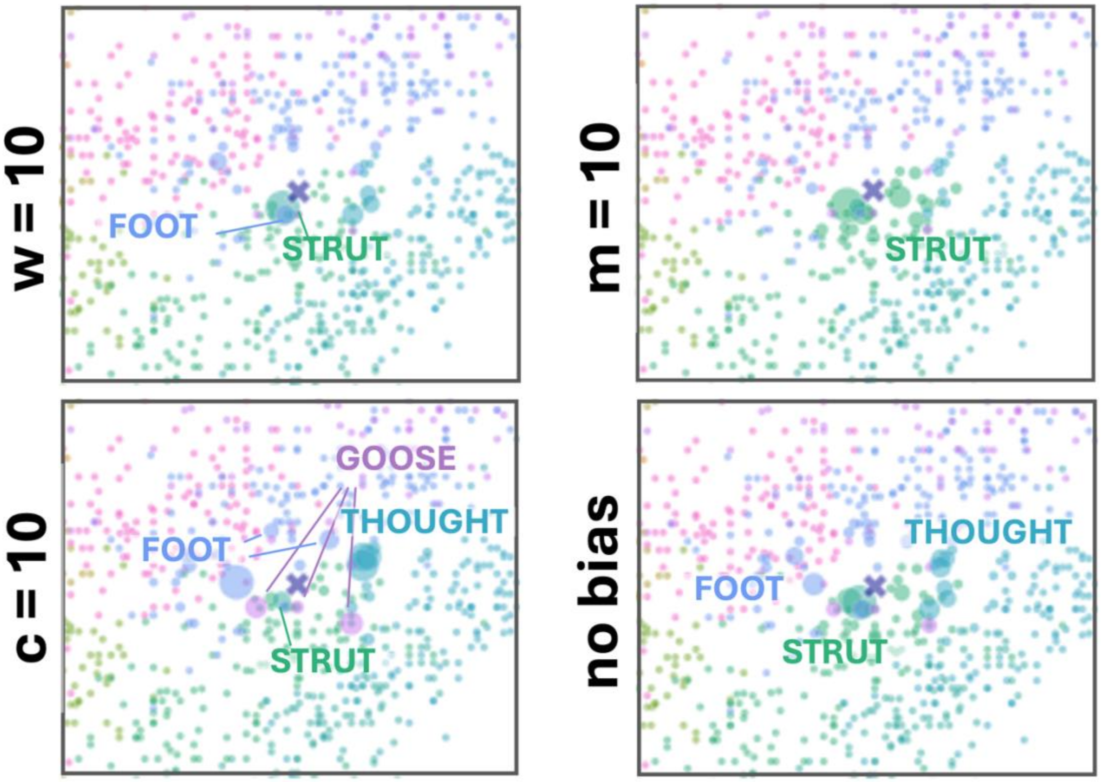


Figure 4.6: Reproduction of Figures 4.4-4.5 insets.

Chapter 5
An experimental investigation of cue types in sociophonetic priming

The group of experiments in speech perception used to argue for social expectation, like those presented in Chapter 3, in fact show significant methodological variation. Before moving on to the methods of this experiment, let's revisit some of these experiments and review variation in their methodologies.

5.1 Variation in sociolinguistic perception methodologies

I've identified three axes of variation for us to consider. First, this research depends heavily on the concept of variation linked to social identity. But what is a social identity within this context? Second, "perception" can be measured in different ways. To what extent are they measuring the same thing? Third, and most relevant to this experiment, how are cues given to listeners?

What is "social information"?

A common locus of identity variation has been geographic-based differences in pronunciation. For example, Niedzielski (1999) investigated Detroit listeners' reaction to Canadian Raising. This feature, in which MOUTH and PRICE have a more mid vowel in the onset, is found in both Michigan and Canadian speech. Hay and colleagues have conducted numerous studies leveraging the relationship between Australian and New Zealand English users (e.g., Hay & Drager 2010; Hay, Nolan & Drager 2006). Another example is Clopper's use of Southern, Northern, and Midland US English, and the regions at which they intersect (2017; see also Clopper, Tamati, and Pierrehumbert 2016; Jones & Clopper 2019). Sumner et al.'s work often leverages the overlap of features across varieties, including arhoticity in both NYC English and Southern Standard British English (Sumner & Samuel 2009; Sumner & Kataoka 2013).

Another factor is gender. In individual and joint work, Johnson and Strand explore gendered expectation. Using non gender-stereotypical voices, Strand found a gradient effect of expectation on sibilant perception (e.g., Strand 1999,

Strand & Johnson 1999, Johnson 2006). Hay et al. (2015) use words that have different frequencies in production according to a binary gender.
Kim and Drager (2017) leverage an age-based difference in Korean stop realization, pairing it with differences in lexical frequency for different age groups. Likewise, Koops et al. (2008) look at the PIN/PEN merger in Houston, Texas. Although the contrast is neutralized for many older speakers in the area, younger residents are less likely to produce the merger.

Race and ethnicity is also considered. Sharese King and Meghan Sumner's 2014 experiment compares African American (Vernacular) English with General American English, which is typically understood to be linked to Whiteness. McGowan (2015) varies the ethnicity of the "speaker" in visual stimuli. This study and others have also considered the imagined speaker's status as an L1 or L2 user of the language (McGowan 2015; Witteman, Weber & McQueen 2014; Gnevsheva 2018).

Using the concept of persona, D'Onofrio has complicated the use of macro-demographic categories such as geographical region, race, and nationality. A Californian in general is compared to a California Valley Girl in particular (D'Onofrio 2015). In a 2019 paper, she investigates the role that personae can play in racialized expectations.

In a perhaps similar vein, some experimenters combine multiple identity categories in the voice as a whole, maximizing the differences between the speakers. For example, Ed King and Meghan Sumner's 2015 paper in *Cognitive Science* compares word association responding to an older black man and a younger white woman. Kapnoula and Samuel (2019) consider the entire voice as "indexical information," varying age, race, gender, and speaking rate. In the experiment, this is tracked primarily by pitch and gender.

How is perception measured?

As another dimension of variation to consider, "perception" is measured in different ways. In part, this seems to depend on the nature of the phonological variation. For example, the shift of a category boundary between SOD and SHOD related to gendered expectation can be measured with a lexical classification task, in which listeners choose which word they heard between alternatives. D'Onofrio (2015) also uses eye-tracking to get on-line measures of perception.

Some tasks seem to target the perception of *sounds* by having listeners match an auditory stimulus to bare vowel reference points along a continuum (e.g., Niedzielski 1999, Hay et al. 2006). Others emphasize lexical access and the perception of *words* through lexical decision tasks (e.g., Sumner & Samuel 2009), in which a participant identifies whether a stimulus is a real word in their language. A quick reaction time indicates that the participant had an easy time accessing that word because of an expectation.

Zheng and Samuel (2019) suggest that some of these tasks may not reflect *perception* per se at all. Rather, this data shows participants' *decision-making* and interpretation. McGowan and Babel (2020) identify two levels of awareness: "high level" and "low level." The concepts of contextual salience, awareness, and attention are incredibly important for this body of work but notoriously hard to define and measure. Differences in conceptualization of "awareness" perseverate into both research questions and results. In this experiment, Central Bolivian listeners performed a discrimination task twice, under two locally significant guises: Spanish monolingual or L1-Quechua bilingual. While Spanish has a 5-vowel system, Quechua is understood to have 3 vowels. Residents are aware of this fact and its reflection in the Spanish of the bilingual Quechua speakers. In this task, expectation was revised midway through the task. For the first set of trials, the expected effect of social expectation was observed. When expectation is revised, the first impression still holds for the low level discrimination task. However, qualitatively, they do adjust expectations at a higher level of awareness.

So, to what extent *are* we conceptualizing and measuring the same phenomenon across these tasks?

How is "social expectation" induced?

The type of cue given to induce an expectation also varies. In general, cues are either visual, textual, or implicit.

One type of visual cue is to show listeners a representation of the "speaker". Some experimenters show participants images or videos of the speaker (Strand 1999; McGowan 2015; D'Onofrio 2019; Koops, Gentry & Pantos 2008; Zheng & Samuel 2019). Others show the participant images of symbolic objects. D'Onofrio uses outlines of the American states California and Michigan, shopping bags and glasses. The visual stimuli in Hay et al. (2015) include stereotypically gendered clothing. Hay and Drager (2010) pull the visual cue into the third dimension by using artifacts. Stuffed toys, a kiwi bird and a koala, represent New Zealand and Australia, respectively.

Another method is to explicitly label the speaker, as D'Onofrio does in a (2018) study comparing Valley Girls and business professionals. Niedzielski (1999) and Hay et al. (2006) include a label on the response sheet and leave the listener to infer an association.

Finally, some studies depend on the listeners' ability to detect social information from features of the voice itself. Social expectation studies work because phonological features are indexically linked to "non-linguistic" social variables. These variables pattern in complex constellations known as indexical fields (Eckert 2008). So, a particular phonological variable is linked to other variables, both structural (e.g., phonological) and social. This is often the case for studies where the experimenter doesn't isolate a particular social variable, as in King and Sumner (2015). In an experiment I presented at ASA and LSA meetings, I asked a related question by crossing socially enregistered syntactic constructions and phonological variants (Remirez 2019, 2020). Taken together, this reflects the integrated nature of language use and perception in interaction, but does leave ambiguous the relationship between the source of information and its effect on perception.

Most relevant for the discussion is the timing of labeling and visual cues that can be observed before the onset of the auditory stimulus in these experiments. The assumption is that these non-speech cues contribute their priming effect *before* speech perception begins. Phonological cues, however, become available at the same time as the stimulus.

A closer look at D'Onofrio (2015)

The experiment in this chapter is directly inspired by Annette D'Onofrio's 2015 paper and is intended as a partial replication and extension of that work. As mentioned above, this work represents an important step toward more accurately capturing the complexity of identity construction. Ethnographic and descriptive work shows that identity is instantiated through interaction (e.g., Bucholtz & Hall 2006), malleable and locally specific. Using non-specific, macro-demographic labels like race or broad geographic regions therefore misses the mark on how language is actually used, how stances are conveyed and positions negotiated during interaction.

Specifically, this experiment invokes the association of the California Vowel Shift with the state in general and the "Valley Girl" persona in particular. Many descriptions of this vowel change exist; I'll follow D'Onofrio and co-authors elsewhere (Podesva et al. 2015) in referring to Eckert's (2004) description. This system of chain shifts includes the lowering of /æ/ as in TRAP to something more like /a/, as in LOT.

Listeners responded to auditory stimuli from two speakers. The target voice was judged by norming participants to be from a White woman in her 30s, whose speech was associated with multiple different regions of the US. In other words, it’s reasonable to assume that listeners would not be surprised to learn that the speaker was from California or not. A distractor voice was a White man in his 30s from the midwest. While hearing the audio, they were presented with icons evoking social factors. A control group saw green or orange circles, while the two test groups saw either a macro-demographic cue, the states of California and Michigan, or a persona cue, a purse and shopping bag or a taped pair of glasses. They received explicit instructions that the glasses speaker had been described as a “nerd” and the shopping bag speaker “a Valley Girl.”

By using eye-tracking, D’Onofrio is able to probe the role of personae in automatic processing. Listeners chose a word from among four orthographic options presented on a screen. Automatic processing can be approximated by measuring looks to each word. For our purposes, we’ll focus on the word classification portion of the experiment.

Word choice for the persona-level cue group was significantly different from the baseline, but not the state-level cue. As predicted, listeners who expected a “Valley Girl” perceived more centralization than those without such expectations.

5.2 Methods

I follow D’Onofrio in leveraging the association between TRAP-backing and California/Valley Girl and trying to measure how much centralization listeners perceive. One reason for this choice is that it’s locally relevant to the San Francisco Bay Area, where this research was conducted.

To simplify the design, I forgo eye-tracking and present each listener with a single voice. Where D’Onofrio focuses on decisions in ambiguous trials, I focus instead on the overall numbers of centralized responses in each condition. The conditions of the experiment design are shown in Table 5.1.

Conditions

The experiment was repeated 4 times (Table 5.2) with slight variations to the stimuli. In each experiment, participants were pseudo-randomly assigned to one of 6 conditions in a 2x3 design (Table 5.1). Implicit cues based on the auditory stimuli guise are crossed with explicit labeling given in the instructions. In Experiments 1-3, listeners were instructed that they would hear a recording of “a Valley Girl,” “a business professional,” or “a person.” Experiment 4 did not include a “person” condition, with listeners assigned to groups 1, 2, 4, or 5. No participants were made aware of the other conditions in the experiment.

		Phonetic cues	
		Creaky voice	**Modal voice**
Instructions	**Valley Girl**	group 1	group 4
	Business professional	group 2	group 5
	Person	group 3	group 6

Table 5.1: Summary of experiment conditions

Stimuli

This experiment uses two sets of 7-point continua, one “professional” guise and one “Valley Girl.” Stimuli creation is schematized in Figure 5.1, and representative stimuli are shown in Figure 5.2.

Experiment	**Participants**	**n**	**Noise (-3dB SNR)**	**Images**
Non-Californians	Anyone on Prolific	84	None	No
Californians	Registered in California	92	None	No
Noise	Registered in California	79	Babble	No
Images	Registered in California	73	Pink	Yes

Table 5.2: Summary of experiments

Undergraduate research assistant Jiacheng Liu used Tandem-STRAIGHT (Kawahara 2008) to resynthesize a 7-point continuum from the words “bat” (TRAP) to “bot” (LOT) based on natural recordings. To get the starting recordings for the stimuli, I recruited self-identified Valley Girls from my extended network to record a word list featuring minimal pairs in “Valley Girl” and “business professional” guises. The words were embedded in the carrier phrase “say [word] again.”

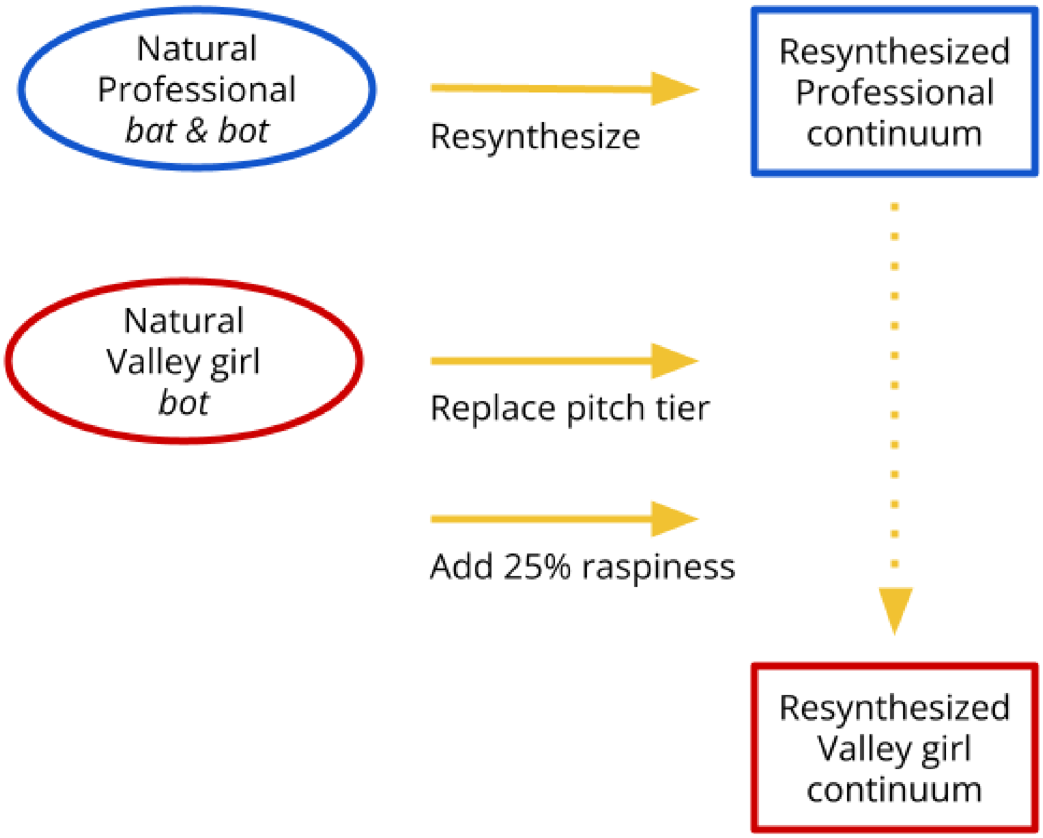
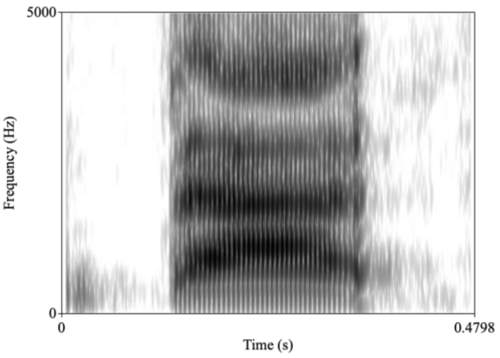
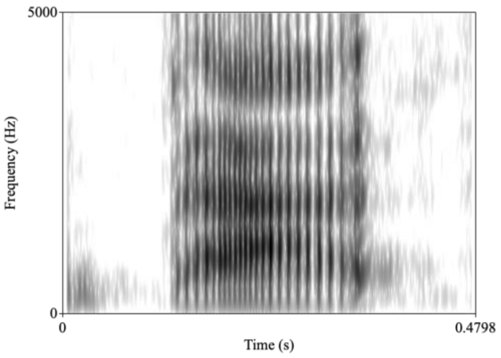
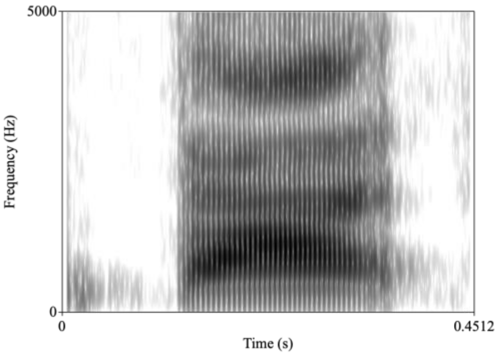
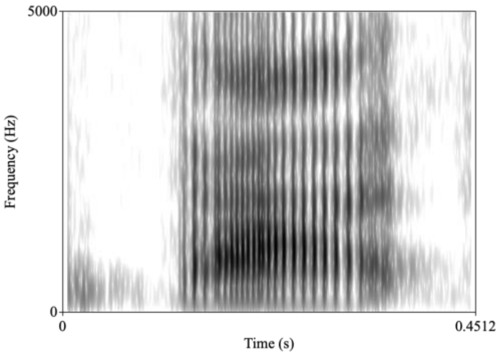


Figure 5.1: Schematization of stimulus creation. Professional guise is shown in dark blue, Valley Girl in red. Ellipses represent the original, natural recordings, while rectangles represent resynthesized continua. Resynthesis processes are shown with solid yellow arrows. The dotted yellow line indicates that the replacement of pitch tier and addition of raspiness were applied to the resynthesized professional continuum.

The speaker whose voice was ultimately used for the stimuli is a non-binary person who self-identified with the labels *woman+* and *woman-adjacent* used in recruiting descriptions. The speaker is in their early 30s and has lived in Southern

California and the San Francisco Bay Area. I recorded them in a sound attenuated booth in the UC Berkeley PhonLab.

Vowels were standardized for length before any resynthesis by duplicating a complete cycle at the midpoint of the vowel, at zero-crossings, until total vowel durations were comparable. As anticipated, the clips recorded in the Valley Girl guise prominently featured creak. Because this made pitch interpolation unreliable, the Valley Girl stimuli were modified from the business professional continuum. The pitch tier was extracted from a Valley Girl recording of "bot" and used to replace that of the business professional continuum. I used the Praat package Vocal Toolkit (Corretge 2012-2023) to add 25% raspiness to each stimulus, approximating the creak in the original Valley Girl recordings. The clips were exported using PSOLA re-synthesis.

Step	**Professional**	**Valley Girl**
1 **bat**		
4		

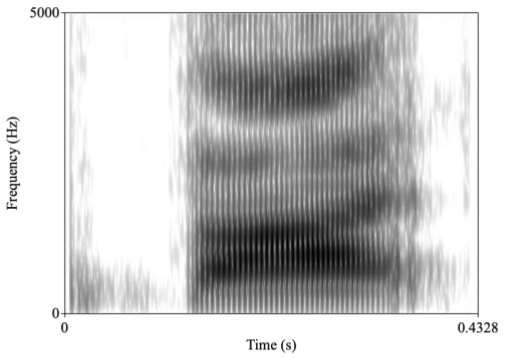
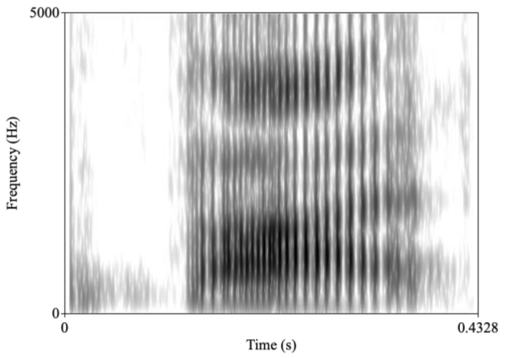
7 **bot**		

Figure 5.2: Steps 1, 4, and 7 of Professional and Valley Girl stimuli continua

The results of a norming task were limited, but of the 6 responses, no one responded that the stimuli were unnatural or not Valley Girl-like. Volunteers listened to 8 clips and rated statements based on how likely they were to be true on a scale of 1-4, with 1 being not at all likely and 4 being very likely. Table 5.3 summarizes responses to the 4 clips belonging to our stimulus talker. It shows the count of 3 ("more likely to be true than false") and 4 ("very likely to be true") responses to some of the statements: Their likelihood of having been described as a "Valley Girl," a "business professional," or "chill"; being from California; or surfing and/or skating. The latter 3 statements are meant to cue qualities associated with the CVS (Podesva 2011).

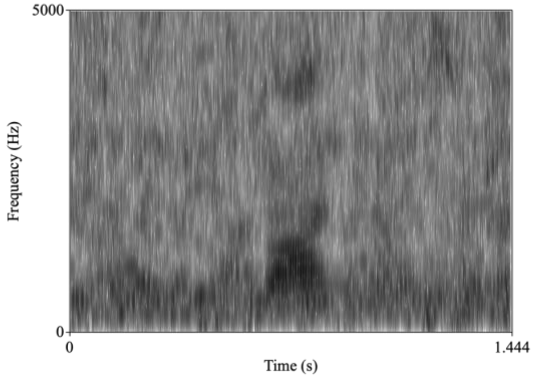
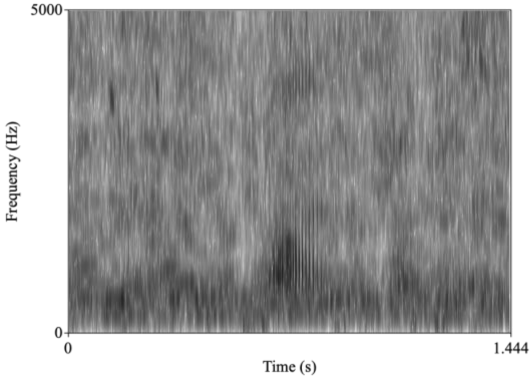
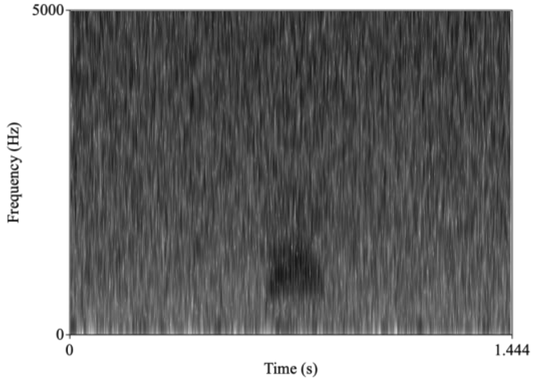
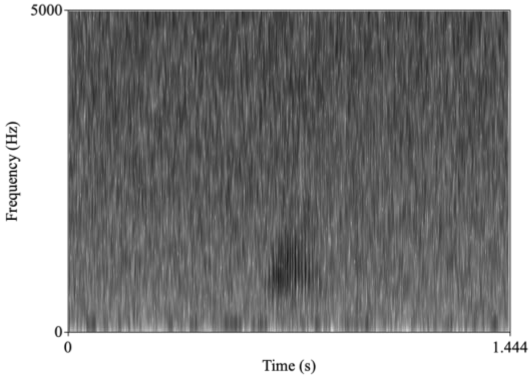
		Number of very/likely to be true responses				
		"Valley Girl"	**"bus. pro."**	**"chill"**	**From California**	**Surfs &/ or skates**
Valley Girl	**TRAP (hat) Re-synth**	2	1	3	3	1
	LOT (bot) Re-synth	4	1	4	4	4
Pro	**TRAP (bat) Re-synth**	1	3	3	3	3
	LOT (hot) Natural	2	2	3	2	1

Table 5.3: Subset of stimuli norming responses

The Valley Girl and professional guises are each more associated with the intended label than with the other. Although only 2 respondents rated the Valley Girl TRAP clip likely to have been described as a Valley Girl, it didn't receive any "not at all likely to be true" ratings.

Embedding stimuli in noise

To increase uncertainty, the stimuli in Experiments 3 and 4 were embedded in noise at a signal-to-noise ratio of -3 dB. First, in Experiment 3, multi-talker babble was used. However, upon closer inspection, a peak in noise overlapped with the ambiguous stimuli, arguably turning this experiment into a phoneme restoration task. To address this, pink noise was used for Experiment 4. Figure 5.3 visualizes the differences between stimuli. Step 5 is used as the example based on the results in Figure 5.6, below, which show an unexpected result around this point.

	Professional	**Valley Girl**
B		
P		

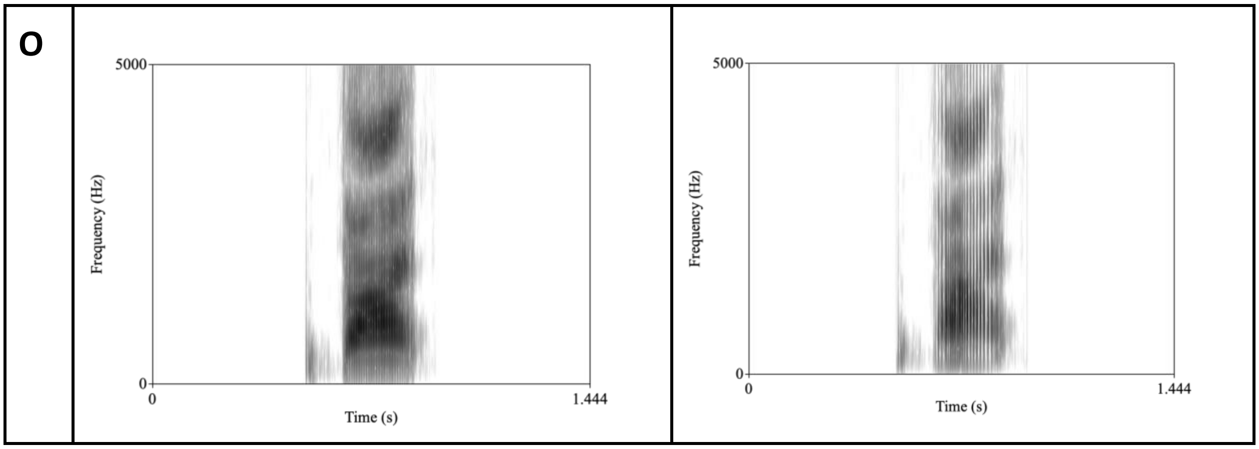


Figure 5.3: Spectrograms of stimuli at point 5. Professional guise is on the left, with Valley Girl on the right. Rows show multi-talker babble (B), pink noise (P), and the original resynthesized continuum, without added noise (O).

Visual stimuli

In addition to the audio stimuli, Experiment 4 included images of women intended to represent a "Valley Girl" and a "business professional" (Figure 5.4). In this condition, the neutral instruction, with "person," was not included. These images were found via searching Google Images.

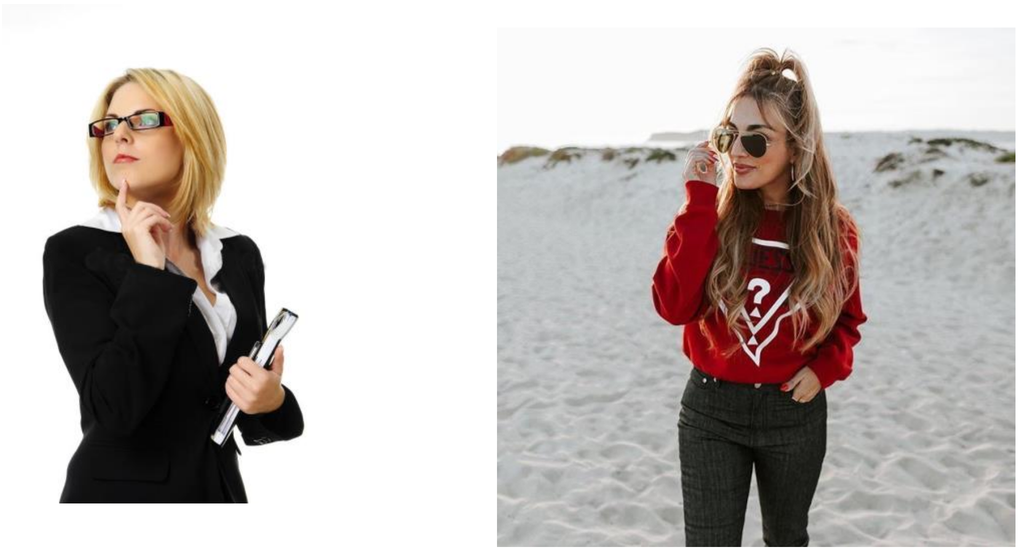


Figure 5.4: Visual stimuli used with business professional (left) and Valley Girl (right) instructions

Task

In this two-alternative forced choice lexical classification task, listeners chose between “bat” and “bot” to label the stimulus. The experiment was compiled for the web using JavaScript by Keith Johnson and administered on Prolific. Participants were limited to those who were registered on Prolific as California residents.

Listeners heard each clip once per trial, with the entire 7-point continuum presented 11 times, for a total of 77 trials. The task was unspeeded, and reaction time was recorded. They used the keyboard to press “z” for “bat” and “m” for “bot.”

Demographic questions

Participants completed a brief questionnaire at the conclusion of the experiment, answering the following questions:

1. How old are you?
   - Drop down of age ranges; 5 year bins for 15-60, 10 year bins for 60+
2. If data are combined into groups based on gender, which group should we include your data with?
   - Drop down: Men, Women, A third group, Do not include my data in any of these groups
3. How would you describe your race or ethnicity?
   - Free response
4. What languages can you use to converse with other people? If more than one, list them in the order of your comfort or proficiency from most to least.
   - Short answer with “English” in gray text
5. Do you consider yourself "online" or engaged in "internet culture" on platforms such as twitter, tiktok, tumblr, or reddit? Do you usually know who "the main character"[1] is? (If you don't know what this question means, answer "No")

[1] A person, usually another poster, who becomes the topic of extended derisive discussion, with their transgressions being treated as common knowledge on the platform.

    - Drop down: Yes; No, but I do use social media; No, I rarely or never use social media

6. Do you think of yourself as a Californian?
    - Drop down: Yes or No

5.3 Participants

Across the 4 experiments, a total of 357 participants were recruited. Demographic details of the latter 3 groups are given below.

Exclusions

In both experiments, participants' data was excluded if they used the same response for over 80% of trials, or if more than 10% of their responses were excluded. Responses were excluded based on reaction time and response. If the reaction time was more than 2 standard deviations outside of that participants' mean. That is, only responses that seemed to be typical for that person were included. A response was also thrown out if an unambiguous endpoint-stimulus was labelled with the inappropriate alternative. If a key other than "z" or "m" was pressed, listeners were instructed to press a valid option. This second response was included, provided that it didn't push the reaction time beyond the cut-off.

Participant demographics

The following sections describe the participants in Experiments 2-4. In phrasing the questions, I considered participants' privacy and self-determination, in addition to my plans for the data. Treating age as a bin at the time of data collection, for example, makes the participant's data less identifiable. The lower need for uniformity in race and ethnicity data led me to prioritize self-determination in a free-response question. The primary role of the demographic data here is to establish coverage of basic demographic categories across the total subject pool.

Age

In all three experiments, participants skew younger, with a mode between 21 and 35. The majority of participants were 40 or younger: about 85%, 87%, and 70%, respectively. Age data are summarized in Table 5.4

	18-20	21-25	26-30	31-35	36-40	41-45	46-50	51-55	56+	Total
Exp 2	11	14	24	16	13	5	5	1	3	92
Exp 3	6	25	18	8	12	5	3	1	1	79
Exp 4	3	14	6	21	7	9	4	4	5	73

Table 5.4: Participant age demographics

Gender

While the gender balance in the noisy-but-no-pictures condition has a relatively even split between men and women, men outnumber women substantially in the other two experiments, with a ratio of 2 (Experiment 4) or 2.5 (Experiment 2) men to each woman participant. Counts are shown in Table 5.5.

In each experiment, the "third group" received a single response. I take this as evidence of two things. First, this category is used by at least one participant. To do ethical research, we must consider the subjective and qualitative experience of volunteers. Because of the way I phrased this question, at least 3 people were not forced to misgender themself during my experiment. Second, this reveals room to improve gender diversity in my sampling and recruiting.

	Men	Women	3rd group	Total
Exp 2	65	26	1	92
Exp 3	39	39	1	79
Exp 4	48	24	1	73

Table 5.5: Participant gender demographics

Race and ethnicity

The free response format of the race and ethnicity question reveals a downside of prioritizing participant self-determination during questionnaires. That is, the responses are difficult to organize. Post-hoc, I divided the responses into the following bins: Asian (including South Asian), Black, Latine, Multiple, or White. Counts are shown in Table 5.6.

	Asian	**Black**	**Latine**	**Multiple**	**White**	**Total**
Exp 2	25	4	20	11	32	92
Exp 3	29	3	14	3	30	79
Exp 4	16	2	8	6	41	73

Table 5.6: Participant ethnicity demographics

While some participants in the “Multiple” category did indicate that they’re Black, there is a glaring underrepresentation of Black participants compared to other groups. Although race was not predictive in the model, this provides important limitations to what we can conclude from the results: We can’t say this is what “people” do if we don’t know about Black people. In no case can we extrapolate totally from individuals to a group, but to do so in this case, when White, Asian, and Latine participants are comparatively so well-represented, would be especially egregious.

Participants in Experiment 3 were the least ethnically diverse. Compared to the other experiments, there are more White participants and fewer Asian and Latine participants. White participants make up 35% and 38% of the first two experiments, respectively, but 56% in the final experiment. This question is especially relevant as we consider possible racialization of the *Valley Girl* persona. That is, when we say “Valley Girl,” do we implicitly mean someone who’s White?

Internet culture

In each experiment, very few participants identified as not being “online” at all, with the majority identifying with “Internet culture.” Responses are summarized in Table 5.7.

	Yes, I’m “online”	Some social media	No social media
Exp 2	64	23	5
Exp 3	46	29	4
Exp 4	48	22	3

Table 5.7: Participant internet use demographics

The question of social media usage was originally designed with the persona label *influencer* in mind. The term influencer was suggested through a response to a social media post as a more modern label associated with this way of languaging. The terms aren’t meant to be synonymous or interchangeable. Rather, the label would leverage the association of the speech style with an online persona that makes use of it. Under this reasoning, engagement with Internet culture could correlate with degree of exposure or familiarity. Although the decision to use the Valley Girl label rather than influencer made the connection to the Internet less relevant, I chose not to remove it from the survey. The question poses negligible risk for participants, but would allow any effect of engagement with the speech style through social media to emerge.

One caveat to this question in any case is that we can’t assume that people are doing the same things online, or that they interpreted the question the same way. I intended to evoke something like a community of practice that takes place online. Although the “community” is loosely defined, there is a shared understanding of “main characters”--those unlucky enough to be the subject of viral critique–and key voices across platforms like TikTok and the late Twitter. However, the phrasing of the question would include someone who interacts with friends and family on Facebook only. Those who don’t participate in the culture may not even be aware it exists, not considering it whatsoever in responding to my question.

5.4 Measuring degree of centralization

This methodology seeks to measure whether the listener expects TRAP to be centralized. That is, will the participant assume, based on other information, that the speaker has features of the California Vowel Shift and accept a more LOT-like vowel as a production of TRAP? Figure 5.5 schematizes the possible results. On the Y-axis, a higher value indicates more “bat” responses, while a lower value indicates more “bot”. On the X-axis is the stimulus number, with step 1, clearly “bat” on the left hand side, step 7 “bot” on the right, and the ambiguous stimuli in the middle.

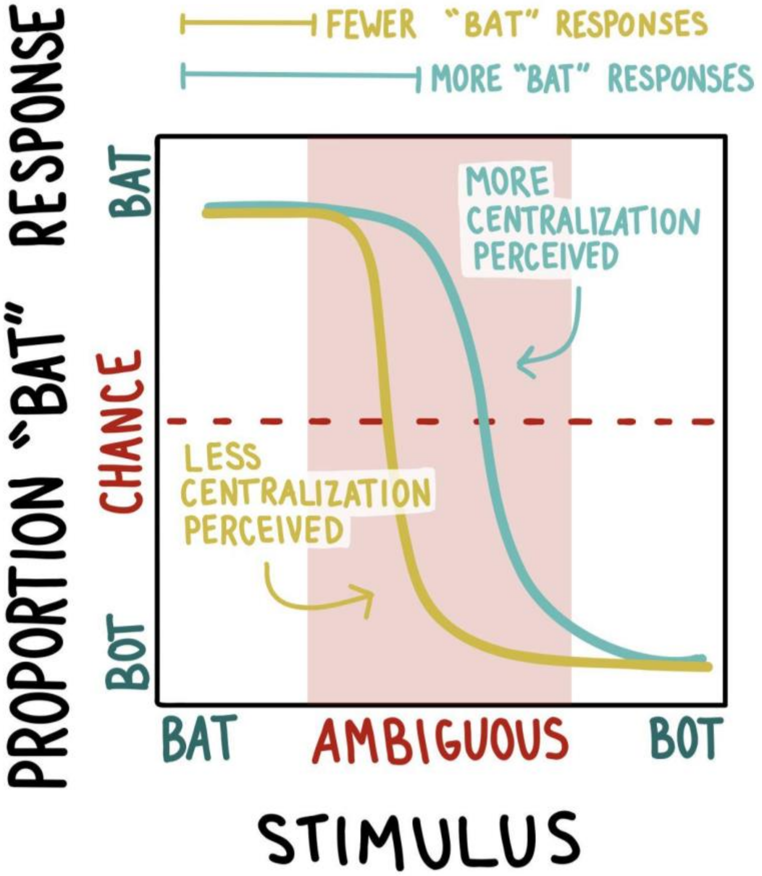


Figure 5.5: Plot schematizing potential results of forced choice experiment with categorical perception boundaries

I interpret a boundary in categorical perception at the point where the sigmoid curve crosses the 0.5 line marking chance. If the curve intercepts the 0.5 line at a later stimulus, as in the blue line, we interpret this as more TRAP-centralization being expected. That is, because TRAP overlaps with LOT in phonetic space, a more central vowel can still be perceived as TRAP. The yellow curve, with a 0.5 intercept earlier in the continuum, represents less tolerance or compensation for a centralized TRAP vowel. To look for an effect, we compare curves for the different conditions of the experiment. The farther right the intercept is, the greater the impact of social expectation. A participant's total number of "bat" responses can be used as an outcome for degree of centralization perceived.

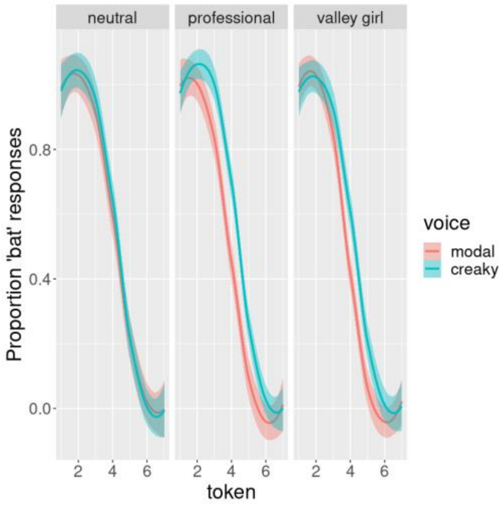
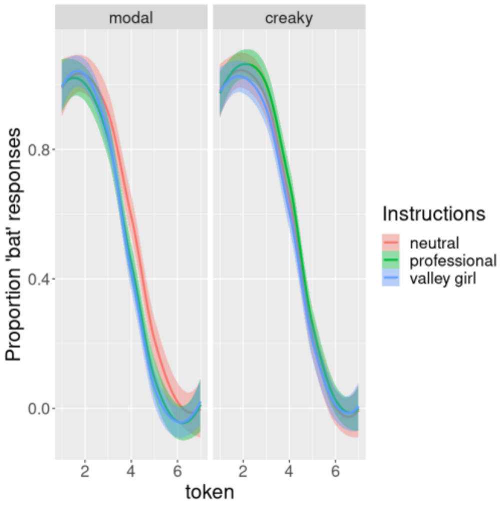
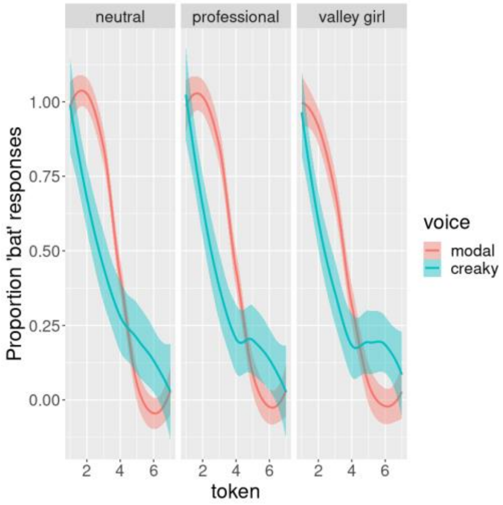
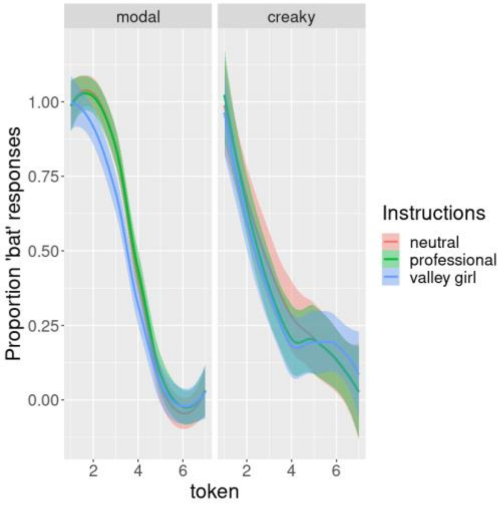
5.5 Results

Classification

The results of this series of experiments are unexpected and fail to answer the initial research question. Recalling the schematization in Figure 5.1, we can qualitatively compare the curves for each experiment (Figure 5.2). In each case, there is no real difference among instruction types. In Experiment 2, there is a small difference between the creaky and modal voice conditions. However, the other experiments show a larger effect in the opposite direction.

The graphs also illuminate an issue with the multi-talker babble stimuli: there is a large burst of noise at roughly the middle of the vowel. In this way, the stimuli almost represented more of a silent center or phoneme restoration manipulation. The consistent pink noise shows the expected smooth curve, rather than the jagged plateau of the variable babble.

The data were submitted to a linear regression analysis in R. Exploratory mixed effects modeling used the lme4 and lmerTest packages. The number of z responses was predicted as a function of phonation and instructions each as main effects, and their interaction. The default levels were the modal voice and the neutral instructions.

<table>
<tr><th></th><th>By Voice</th><th>By Instruction</th></tr>
<tr><td><b>Exp 2</b>
<b>No noise</b></td><td>
</td><td>
</td></tr>
<tr><td><b>Exp 3</b>
<b>Babble</b></td><td>
</td><td>
</td></tr>
</table>

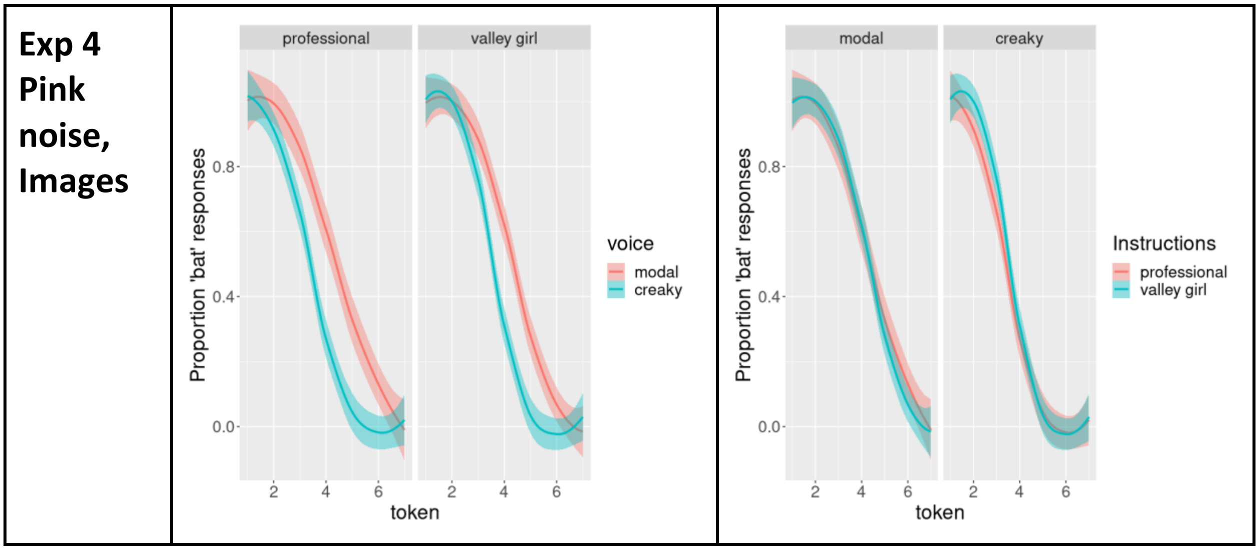


Figure 5.6: Results of classification task. Graphs plot the proportion ‘bat’ response against the token number, with ‘bat’ at token 1 and ‘bot’ at token 7. Rows show results for different experiments. For each experiment, both ways of grouping conditions are shown: by voice, with instructions separated into subplots (left) and by instruction, with different subplots for voice (right).

The regression models corroborate the qualitative observations, consistently showing no effect of instruction type, even with the inclusion of pictures. Further, we assumed that a creaky voice would be associated with a Valley Girl guise, and therefore more centralization. We in fact found the opposite. In the two experiments including noise, there is a significant main effect of phonation type, with creaky voice’s negative coefficient indicating less centralization. Table 5.8 shows a closer look at the regression for the final experiment ($p < 0.001$, adjusted $R^2 = 0.319$).

Reaction time

During the experiment, reaction time was recorded in milliseconds. The duration of clip was subtracted from this number to get reaction time from offset of stimulus in milliseconds. These values were log-transformed for analysis. This analysis also considers the step along the continuum or token number. We centered this number around 0 and took the absolute value. So, 0 represents the most ambiguous stimulus, the midpoint of the continuum, with higher values indicating stimuli closer to the endpoints.

Call:					
lm(formula = Number of z responses ~ Voice condition * Instruction condition)					
Residuals:					
	Min	**1Q**	**Median**	**3Q**	**Max**
	-12.4444	-3.4444	-0.8125	4.1875	16.0526
Coefficients:					
	Estimate	**Std. Error**	**t value**	**Pr(>|t|)**	
(Intercept)	42.812	1.57	27.271	< 2e-16	***
Voice-Creaky	-10.762	2.106	-5.11	2.74E-06	***
Instructions-Val	-1.368	2.158	-0.634	0.528	
Voice-Creaky:Instructions-Val	4.265	2.95	1.446	0.153	
Residual standard error	6.279 on 69 degrees of freedom				
Multiple R-squared	0.3746		Adjusted R-Squared:		0.3192
F-statistic	12.25 on 3 & 69 DF		p-value:		1.618e-06

Table 5.8: Regression table for classification model, Experiment 4.

I modeled log reaction time using a mixed effects linear regression, using lme4 and lmerTest, with fixed effects token, trial, and instructions * voice, and random intercepts for listener. The results in Table 5.9 show that there is no effect of experiment condition on reaction time. Effects of token and trial are both significant (p < 0.001). As expected, participants take longer on ambiguous stimuli and respond quicker as the experiment progresses.

Call:

lmer(formula = log rt ~ token + trial + Voice condition * Instruction condition + (1|Listener))

Scaled residuals:						
	Min	**1Q**	**Median**	**3Q**	**Max**	
	-10.6875	-0.5618	-0.1360	0.3367	7.727	
Random Effects:						
	Groups	Name	Variance	Std.Dev.		
	Listener	(Intercept)	0.04286	0.2070		
	Residual		0.02901	0.1703		
Fixed effects:						
	Estimate	**Std. Error**	**df**	**t value**	**pr(>|t|)**	
(Intercept)	7.44E+00	5.23E-02	7.05E+01	142.305	< 2e-16	***
token	-3.49E-02	4.43E-03	5.48E+03	-7.895	3.47E-15	***
trial	-2.48E-03	1.03E-04	5.48E+03	-24.054	< 2e-16	***
Instructions-Val	9.35E-04	7.15E-02	6.90E+01	0.013	0.99	
Voice-Creaky	-3.28E-02	6.98E-02	6.90E+01	-0.47	0.64	
Instructions-Val:Voice-Creaky	-6.77E-02	9.77E-02	6.90E+01	-0.693	0.491	

Table 5.9: Regression table for reaction time model, Experiment 4.

Why was there no pattern without noise?

One reason to add noise to perception experiments is to increase the *uncertainty* listeners experience. In Regier and Xu's (2017) proposal, perception can be modeled as the combination of general categories and specific stimuli. When perceivers are *uncertain*, they draw more heavily from the less specific category representation. The effect of this combination is similar to resonance. The response leans heavily towards typical category members, blurring the details of a specific stimulus with ideas about the category. Perhaps in Experiment 1, without noise to increase uncertainty about the stimulus, listeners drew less from ideas about categories.

Why is there no effect of labeling in instructions?

One reason there is no effect of which instructions the participants saw may be that the cue was too subtle. The task was completed online, without a researcher present to ensure instructions were read carefully. Although the label was bolded and repeated on the page for each trial, it's possible that some participants only skimmed the instructions and focused on the sound clip.

For a similar reason, we expected the effect to be more robust with visual cues added. This was not the case. The images were not normed on the same population for their associations, so it's possible that the images don't evoke the intended personae for these listeners. However, the effect, or lack thereof, is consistent with or without the images, suggesting that a mismatch between label and visual stimuli is not an issue.

We might also ask how relevant the term "Valley Girl" is for younger speakers. I most commonly hear the term used to critique the behavior of young, typically White or White-adjacent, women or woman-adjacent people. Media openly mocks their "vocal fry" (creaky voice) and "uptalk" (appellate intonation), and associates these features with negative traits such as being immature, unserious, and insincere. Even if the persona is still salient, would a different, more modern label be more evocative for younger listeners?

The label *influencer* was suggested by Cassandra Jacobs (p.c.) as a replacement for a persona associated with this way of languaging. As is so often the case when we do research on human beings, how language users construct the meaning of *influencer* or *Valley Girl* and understand that term could be exploded into its own research program. One reason I chose to follow D'Onofrio in using *Valley Girl* was simply that a choice needed to be made. Without a systematic understanding of the terms based in ethnographic and descriptive insights, there was no clear path that one alternative was superior. When in doubt, replicate.

Another reason is that *influencer* in fact encompasses a wide range of personae. While the prototypical influencer may still be Kim Kardashian, this term can also be interpreted as an occupation, rather than a persona. Whether different terms would affect the results is really an empirical question that can be studied in follow-up experiments.

Finally, an assumption throughout this dissertation is that cues take perceptual precedence over each other given context and expectation. We intended to create a clear cue to persona with labeling and images, inducing social expectation. In terms of the GCM, base activation would be increased for each exemplar that's been categorized as "Valley Girl." Because many of these exemplars would also contain TRAP-centralization, those centralized vowels would contribute more activation to categorization. This could have still happened, but the effect is masked. Recall the discussion from Chapter 2 addressing frequency effects. Multiple factors go into similarity and activation calculation; even though frequency plays a role, categorization is still fundamentally about similarity. If unscaled similarities already point to a clear percept, the differences in resting activation won't be observable.

A future version of this experiment should use more ambiguous auditory stimuli. Uncertainty was induced by embedding speech in noise, but this may not have been enough uncertainty. The reaction time data, which show that tokens near the middle of the continuum took longer to respond to, support that these stimuli were ambiguous. But were they ambiguous enough? I chose to instantiate the California Vowel Shift with the words "bat" and "bot" to follow D'Onofrio's (2015) "sack" and "sock." Trying pairs of vowels that are generally more confusable should show a more extreme effect. If the vowels are more likely to be confused without the priming, there's more room for the difference in base activation due to priming to play a role. The difference in phonological contexts may also change perception of the vowel quality. Further, I'm curious about the role of chain shifting in perception of these vowels. What would it look like to do a 3-alternative forced choice, for example, adding a "bet" to "bot" continuum? Another change would be to mix in a filler voice, as D'Onofrio did. Asking listeners to make a comparison between two voices may likewise enhance the importance of category-level cues over the specific stimulus.

Why is creaky voice associated with less centralization?

One of the unexpected findings of this study is that the two phonetic features expected to resonate–creaky voice and centralization–are not in fact associated.

Listeners who were in a creaky voice condition in fact responded with fewer "bat"s than their modal voice counterparts.

One explanation is that the creaky vowel simply doesn't carry enough spectral information. Stimuli were normalized for vowel duration before creak was added, but this modification dropped the number of vowel quality-bearing glottal pulses in each clip. The question then becomes why this lack of information led to more "bot"s. It also bears repeating that the norming done before the experiment received limited responses. I continued with the stimuli in large part in the interest of time.

Another idea is that by adding creak and changing the intonation to create a second continuum, we changed the stimuli in unexpected ways. These changes were meant to invoke the Valley Girl persona, but they could have shifted the boundary in and of themselves. If this effect is robust enough, it may overshadow any sociolinguistic associations. For example, could we be inducing perceptual compensation for coarticulation? Listeners are familiar with the predictable acoustic effects segments have on each other during speech production. Expecting this variation, listeners may attribute consistent acoustic cues to coarticulation, rather than another source.

Another explanation is that the task caused people to be mindful of their biases. Modern listeners could be actively countering negative stereotypes about young women and resisting status quo interpretations of "appropriateness." While this would be a nice story, it seems unlikely that such an effect would be consistent across participants.

How does this relate to the simulations?

When I designed this experiment, I imagined a neat set of data which I could compare to the output of modeling. The effect of vocal guise would be modeled via resonance, while the effect of instructions would be modeled with an increase in resting activation level. If the steps in the simulation can approximate the behavioral effects, that would support the corresponding theoretical account. The "bones" of this experiment and its connection to the type of simulations in the previous chapter remain ready for analysis once the relevant data can be gathered. As discussed above, the effect may be more observable with more phonetically ambiguous or socially representative stimuli, or a new set of personae and enregistered phonetic features.

Chapter 6
Conclusion

In the introduction, I quote Whorf's famous description of the world as a "kaleidoscopic flux of impressions" (1940). I cannot read this phrase without picturing my undergraduate thesis advisor, Michel Achard, gesturing expansively in front of a projector in Rice University's Baker Hall. In those lectures I learned concepts like fuzzy boundaries and prototypes, internalizing some basic ideas about categorization that would underlie the rest of my research. All cognition is fundamentally categorization, assigning a label to something so that we can conceive of it. Even insects sort things into categories: food or not food. The categories of our language don't constrain what we can perceive, but they do shape our mental representations.

Humans are presented with an absolutely overwhelming array of stimuli. We parse this variation into clusters and construct metaphorical boundaries around them. The category boundary separates phenomena, creating the possibility of a binary sameness or difference. The act of perception sorts novel stimuli into these categories. We expect the things we encounter in the world to share properties with other examples of the "same." Sociophonetic priming results pulling on these expectations demonstrate a link between phonetic and social categories. This raises the question: How does our knowledge of the "high-level" stuff integrate with our perception of "low-level" phonetic detail?

The more I tried to answer this question, the more I was drawn to a secondary one: How do we as researchers *understand* the relationship? For many sociolinguists, the answer seemed to be "Exemplar Theory." This term itself muddies the water: although it can be used to refer to a number of distinct models, it gets presented as a single theory. The emphasis on experiences, multimodality, and flexible categories make episodic models a natural choice for describing sociophonetic perception. At the same time, a tendency to de-emphasize or underspecify the particulars of the model can produce confusion in how behavior relates to underlying phonetic processing. To tackle these two interrelated questions, my first step was to clarify what "Exemplar Theory" is, and to choose a particular model to investigate further.

In Chapter 2, I argue for the necessity of precision when referring to an episodic model of speech perception. Exemplar models tend to have 3 major propositions in common: Humans store memories of past linguistic experiences; these exemplars are linked to categories; and new stimuli are categorized on the basis of comparison to stored exemplars. Scholars have proposed a distinction between Syntactic and Phonetic Exemplar theory (Hay & Bresnan 2006, Bod & Cochran 2007), capturing a split between models like Construction Grammar or Data-Oriented Parsing, which are used to describe syntactic structure, and models used to describe speech. The set of episodic models, even within perception, contains contradictory assumptions. If we're to interpret results within the light of "exemplar theory," which assumptions should we take up? Being specific and explicit about the mechanism enables more accurate synthesis of ideas and results from different researchers' work. Examining variation across a set of models is not only clarifying but generative. Comparing one model to another, noting their differences, establishes the crucial aspects of each. Further, a clear understanding of these differences provides a framework for asking deeper, more specific questions, filling the gaps left by incremental scientific progress.

In particular, I elaborate the Generalized Context Model, or GCM (Nosofsky 1986). The GCM is an extension of Medin and Schaffer's context theory of classification (1978). In this model, similarity between each exemplar and a novel stimulus is calculated based on Euclidean distance, with a variable attention weight scaling distance along each dimension of the exemplars. Activation is calculated based on this similarity, weighted by each exemplar's base activation level. The evidence for category membership is the sum of activation values for each exemplar belonging to that category. The percept will be whichever category has the most activation, relative to the total amount of activation in the system. Frequency of a category influences categorization: A more common category will have more exemplars to contribute evidence for category membership. Categorization can also be skewed by the relative weighting of the dimensions. Different sets of weights will increase or decrease distance, therefore changing the amount of activation within each category. Likewise, at the exemplar level, base activation can be raised, or lowered, changing the amount of evidence for its category.

I compare the GCM to dual-route or "difference in encoding" accounts, as described by Sumner et al. (2014). The dual-route approach is embraced for its ability to account for stable effects in perception, like social prestige. The GCM, however, does not allow for this type of difference in storage. I propose two processes that could achieve a similar result, and built the program in Chapter 4 to explore these mechanisms. First, I present the resting activation level of each exemplar as an underutilized vector for sociolinguistic effects. The stable robustness of representations of socially salient speech, regardless of frequency, could build up from repeated priming to exemplars, rather than as the result of differences at the time of encoding. Second, I reintroduce resonance as a component of the GCM, following on Johnson's (2006) proposal. During a speech perception event, listeners form an initial impression of the category a stimulus likely belongs to. Based on this preliminary categorization, activation spreads to other exemplars within that category. In this way, a "top-down" effect of social expectation can be generated "bottom-up" from the stimulus.

In Chapter 3, I explore some of the sociophonetic perception results these models need to account for. For example, Niedzielski's 1999 study in Detroit, Michigan, found that listeners responded differently to identical stimuli based on whether the label associated the clip with Canada or Michigan. This result has been replicated with New Zealand and Australian English (Hay, Nolan, & Drager 2006). The effects appear to be multimodal, with visual stimuli exerting a similar priming influence on perception (e.g., Strand & Johnson 1999, McGowan 2015). I go on to review the variation in how social categories are considered and cued, previewing motivations for the experiment in Chapter 5. These experiments vary in the type of social category, the way the category expectation is induced, and in how perception is measured. Where Chapters 2 and 3 examine variation in episodic modeling and sociophonetic priming literature, Chapters 4 and 5 address some of the questions this exercise raises.

Chapter 4 introduces the Python library ExemPy, built to simulate speech perception events according to the Generalized Context Model. Simulations of speech perception are relevant for both empirical and theoretical applications. First, there may be cases where a perception result is useful but not possible to obtain. In these situations, ExemPy can provide a well-motivated hypothesis that scaffolds future investigations. Additionally, the simulation allows us to visualize what perception would look like if different accounts are true, answering questions that may be difficult to tease apart from experimental evidence alone. These findings can, in turn, motivate future experiments.

The ExemPy library takes a dataframe of "exemplars," where each row is an observation. Among the columns should be at least one category and at least one dimension of variation. Attention weights can be supplied by the researcher, for example, to explore the impact of cue weighting; or, they can be estimated using parameter optimization. I use Peterson & Barney's famous 1952 dataset of American English vowels to simulate two common types of perception experiment: forced choice and identification. Compared to Hillenbrand and Gayvert's experimental results (1993), the confusion matrices for ExemPy are highly correlated. In collaborative work with Dr. Sarah Bakst, we used ExemPy to explore altered acoustic feedback data. In Bakst's experiment, listener's heard a re-synthesized version of the vowels they produced. To counter the perturbation, listeners will adjust their production to varying degrees. Previous work (Niziolek et al. 2013) used listeners' perceptual category boundaries to analyze their production data. In this work, we asked whether we can reap a similar benefit to analysis by estimating these boundaries, enabling new ways to explore the data. Finally, I present ExemPy as a tool to understand resonance and priming as mechanisms for sociophonetic perception. Priming is modeled as an increase to base activation before any comparison between exemplars and stimuli. Resonance, however, happens after activation and evidence have already been calculated. Activation spreads among exemplars belonging to likely categories, solidifying expectations from initial impressions. I explore this proposed difference in timecourse in the next chapter.

Chapter 5 reports the results of a two-alternative lexical classification task. Returning to the type of behavioral evidence presented in Chapter 3, I identify three key areas of variation across studies: First, what constitutes a social identity? Second, what task is used to measure perception? Finally, how is social expectation induced? Specifically, some researchers cue the social identity separately from the voice, whether that's with labels (e.g., D'Onofrio 2018) or visual stimuli (e.g., McGowan 2015). Other studies depend on the listener to infer social characteristics from the voice itself (e.g., King & Sumner 2015). I map this distinction in methodology to the one between priming and resonance. Priming, like labeling or visual stimuli, sets the expectation before the stimulus to be categorized has been heard. Inferences based on the voice, however, can only occur once categorization has already begun.

To draw this distinction out, I designed a lexical classification experiment using both types of cues. Following D'Onofrio (2015), I used the labels "valley girl" and "business professional," and the association of the valley girl persona with the California Vowel Shift (CVS). For a speaker participating in the CVS, the vowel in "bat" will be centralized to sound more like "bot." I used the number of "bat" responses as a proxy for the degree of centralization, or TRAP-backing, perceived.

Listeners in the experiment heard a continuum of "bat" to "bot," with or without creak, an enregistered phonetic feature associated with valley girls. The instructions indicated that the participant was going to hear either a valley girl or a business professional; in the final version of the experiment, they also saw a picture representing that persona. The results of the classification task are unexpected given the literature, and fail to address the original research question. Unlike D'Onofrio, I did not find an effect of persona label on the degree of centralization perceived. Likewise, where I expected the creaky voice stimuli to be associated with less centralization, it was in fact the boundary for the modal voice that showed a more centralized "bat." To gather data that fits into the proposed simulations, I suggest two changes for follow-up experiments. First, the ethnographic relevance of the persona label, and their associations with phonetic features, should be considered more carefully and developed more empirically. Second, increasing uncertainty in the experiment would allow category effects more space to become visible.

In this dissertation, I've raised theoretical, empirical, and methodological questions about categorization, social identity, and the perception of spoken language. Within the Generalized Context Model and similar episodic models, we can understand priming and activation spreading as depending on the representational "sameness" of category members. This results in sociophonetic priming effects where the same stimulus can be perceived differently in different contexts. At a more meta level, how does our understanding of models and types of evidence as "the same" influence our research? My experiment establishes a design for empirically considering differences in the literature.

Bibliography

Ames, H., & Grossberg, S. (2008). Speaker normalization using cortical strip maps: a neural model for steady-state vowel categorization. *The Journal of the Acoustical Society of America*, *124*(6), 3918-3936.

Barreda S (2015). *phonTools: Functions for phonetics in R.*. R package version 0.2-2.1.

Bod, R. (2006). Exemplar-based syntax: How to get productivity from examples. *The linguistic review*, *23*(3), 291-320.

Bod, R., & Cochran, D. (2007). Introduction to exemplar-based models of language acquisition and use. In *Proceedings of the ESSLLI Workshop "Exemplar-Based Models of Language Acquisition and Use"', ESSLLI*.

Bucholtz, M., & Hall, K. (2005). Identity and interaction: A sociocultural linguistic approach. *Discourse studies*, *7*(4-5), 585-614.

Clopper, C. G. (2017). Dialect interference in lexical processing: Effects of familiarity and social stereotypes. *Phonetica*, *74*(1), 25-59.

Clopper, C. G., Tamati, T. N., & Pierrehumbert, J. B. (2016). Variation in the strength of lexical encoding across dialects. *Journal of phonetics*, *58*, 87-103.

Corretge, R. (2012-2023). Praat Vocal Toolkit. https://www.praatvocaltoolkit.com

D'Onofrio, A. (2015). Perceiving personae: Effects of social information on perceptions of TRAP-backing. *University of Pennsylvania Working Papers in Linguistics*, *21*(2), 5.

D'Onofrio, A. (2019). Complicating categories: Personae mediate racialized expectations of non-native speech. *Journal of Sociolinguistics*.

Diehl, R. L., Lotto, A. J., & Holt, L. L. (2004). Speech perception. *Annu. Rev. Psychol.*, *55*, 149-179.

Eckert, P. (2004). California vowels. Radio interview on *All Things Considered*, February 24. http://www.stanford.edu/~eckert/vowels.html

Eckert, P. (2008). Variation and the indexical field 1. *Journal of sociolinguistics*, *12*(4), 453-476.

Englebretson, R. (2007). Stancetaking in discourse: An introduction. *Stancetaking in discourse: Subjectivity, evaluation, interaction*, *164*, 1-26.

Englebretson, R. (2023). Epilogue: commentary on stancetaking in motion. *Text & Talk*, *43*(5), 721-731.

Foulkes, P., & Docherty, G. (2006). The social life of phonetics and phonology. *Journal of Phonetics*, 34(4), 409-438.

Fowler, C. A. (1986). An event approach to the study of speech perception from a direct-realist perspective. *Status Report on Speech Research, edited by IG Mattingly and N O'Brien, Haskins Laboratories, New Haven, CT*, 139-169.

Gahl, S., & Garnsey, S. M. (2004). Knowledge of grammar, knowledge of usage: Syntactic probabilities affect pronunciation variation. *Language*, 748-775.

Ganong, W. F. (1980). Phonetic categorization in auditory word perception. *Journal of experimental psychology: Human perception and performance*, *6*(1), 110.

Gnevsheva, K. (2018). Variation in foreign accent identification. *Journal of Multilingual and Multicultural Development*, *39*(8), 688-702.

Goldberg, A. E. (2006). *Constructions at work: The nature of generalization in language*. Oxford University Press on Demand.

Goldinger, S. D., & Azuma, T. (2003). Puzzle-solving science: The quixotic quest for units in speech perception. *Journal of Phonetics*, *31*(3-4), 305-320.

Goldinger, Stephen D. (1998). "Echoes of echoes? An episodic theory of lexical access." *Psychological review* 105(2).

Hay, J. & Drager, K. (2010). Stuffed toys and speech perception. *Linguistics*, 48(4), pp. 865-892. Retrieved 11 Sep. 2019, from doi:10.1515/ling.2010.027

Hay, J., & Bresnan, J. (2006). Spoken syntax: The phonetics of giving a hand in New Zealand English. *The Linguistic Review*, *23*(3), 321-349.

Hay, J., Drager, K., & Warren, P. (2010). Short-term exposure to one dialect affects processing of another. *Language and speech*, *53*(4), 447-471.

Hay, J., Walker, A., Sanchez, K., & Thompson, K. (2019). Abstract social categories facilitate access to socially skewed words. *PloS one*, *14*(2).

Hay, J., Nolan, A. & Drager, K. (2006). From fush to feesh: Exemplar priming in speech perception. The Linguistic Review 23(3). 351–379.

Hillenbrand, J., & Gayvert, R. T. (1987). Speaker-independent vowel classification based on fundamental frequency and formant frequencies. *The Journal of the Acoustical Society of America*, *81*(S1), S93-S93.

Hintzman, D. L. (1986). " Schema abstraction" in a multiple-trace memory model. *Psychological review*, *93*(4), 411.

Johnson, K. (1997). Speech perception without speaker normalization: An exemplar model. *Talker variability in speech processing*, 145-165.

Johnson, K. (2006). Resonance in an exemplar-based lexicon: The emergence of social identity and phonology. *Journal of phonetics*, *34*(4), 485-499.

Jones, Z., & Clopper, C. G. (2019). Subphonemic Variation and Lexical Processing: Social and Stylistic Factors. *Phonetica*, *76*(2-3), 163-178.

Kapnoula, E. C., & Samuel, A. G. (2019). Voices in the mental lexicon: Words carry indexical information that can affect access to their meaning. *Journal of Memory and Language*, *107*, 111-127.

Kim, J., & Drager, K. (2017, January). Sociophonetic Realizations Guide Subsequent Lexical Access. In *INTERSPEECH* (pp. 621-625).

King, E., & Sumner, M. (2015). Voice-specific effects in semantic association. In *CogSci*.

King, S., & Sumner, M. (2014). Voices and variants: Effects of voice on the form-based processing of words with different phonological variants. In *Proceedings of the Annual Meeting of the Cognitive Science Society* (Vol. 36, No. 36).

Koops, C., Gentry, E., & Pantos, A. (2008). The effect of perceived speaker age on the perception of pin and pen vowels in Houston, Texas. *University of Pennsylvania Working Papers in Linguistics: Selected papers from NWAV 36* 14: 91–101.

Kruschke, J. K. (2008). Models of categorization. *The Cambridge handbook of computational psychology*, 267-301

Mattingly, I. G., Liberman, A. M., Edelman, G. M. G., Gall, W. E., & Cowen, W. M. (1988). Auditory Function: Neurological Bases of Hearing.

McGowan, K. B. (2015). Social expectation improves speech perception in noise. *Language and speech*, 58(4) 502–521.

McGowan, K. B., & Babel, A. (2020). Perceiving isn't believing: Divergence in levels of sociolinguistic awareness. Language in Society, 49(2), 231-256.

Medin, D. L., & Schaffer, M. M. (1978). Context theory of classification learning. *Psychological review*, *85*(3), 207.

Mendoza-Denton, N. (2007). Sociolinguistic extensions of Exemplar Theory. In J. Cole & J. I. Hualde (Eds.) *Papers in Laboratory Phonology 9*. Berlin: Mouton de Gruyter.

Mendoza-Denton, N. (2014). *Homegirls: Language and cultural practice among Latina youth gangs*. John Wiley & Sons.

Munson, B. (2010). Levels of phonological abstraction and knowledge of socially motivated speech-sound variation: A review, a proposal, and a commentary on the papers by Clopper, Pierrehumbert, and Tamati, Drager, Foulkes, Mack, and Smith, Hall, and Munson. *Laboratory Phonology*, *1*(1), 157-177.

Munson, B., & Solomon, N. P. (2004). The effect of phonological neighborhood density on vowel articulation. In *Speech, Language, and Hearing Research*, 47, 1048-1058.

Niedzielski, N. (1999). The effect of social information on the perception of sociolinguistic variables. *Journal of language and social psychology*, *18*(1), 62-85.

Niziolek, C. A., & Guenther, F. H. (2013). Vowel category boundaries enhance cortical and behavioral responses to speech feedback alterations. *Journal of Neuroscience*, *33*(29), 12090-12098.

Nosofsky, R. M. (1986). Attention, similarity, and the identification–categorization relationship. *Journal of experimental psychology: General*, *115*(1), 39.

Pierrehumbert, J. B. (2003). Phonetic diversity, statistical learning, and acquisition of phonology. *Language and speech*, 46(2-3), 115-154.

Podesva, R. J. (2011). The California vowel shift and gay identity. *American speech*, *86*(1), 32-51.

Podesva, R. J., D'onofrio, A., Van Hofwegen, J., & Kim, S. K. (2015). Country ideology and the California vowel shift. *Language Variation and Change*, *27*(2), 157-186.

Regier, T., & Xu, Y. (2017). The Sapir-Whorf hypothesis and inference under uncertainty. *Wiley Interdisciplinary Reviews: Cognitive Science*, *8*(6), e1440.

Remirez, E. (2020). “Interacting phonetic and syntactic cues in perception.” [Poster presentation.] Linguistic Society of America meeting 2020. New Orleans, Louisiana, USA.

Remirez, E., & Bakst, S. (2023). *Leveraging exemplar modeling of vowel perception to explore variability in adaptation to altered auditory feedback*. [Manuscript submitted for publication.]

Remirez, E., & Johnson, K. (2021). “Activation as social ‘encoding’ in a Python implementation of phonetic exemplar theory.” [Poster presentation.] Linguistic Society of America meeting 2021.

Sanchez, K., Hay, J., & Nilson, E. (2015). Contextual activation of Australia can affect New Zealanders' vowel productions. *Journal of Phonetics*, *48*, 76-95.

Silverstein, M. (2003). Indexical order and the dialectics of sociolinguistic life. *Language & communication*, *23*(3-4), 193-229.

Strand, E. A. (1999). Uncovering the role of gender stereotypes in speech perception. *Journal of language and social psychology*, *18*(1), 86-100.

Strand, E. A. (2000). *Gender stereotype effects in speech processing* (Doctoral dissertation, The Ohio State University).

Strand, E. A., & Johnson, K. (1996, October). Gradient and visual speaker normalization in the perception of fricatives. In *KONVENS* (pp. 14-26).

Sumner, M. (2015). The social weight of spoken words. *Trends in Cognitive Sciences*, *19*(5), 238-239.

Sumner, M., & Kataoka, R. (2013). Effects of phonetically-cued talker variation on semantic encoding. *The Journal of the Acoustical Society of America*, *134*(6), EL485-EL491.

Sumner, M., Kim, S. K., King, E., & McGowan, K. B. (2014). The socially weighted encoding of spoken words: A dual-route approach to speech perception. *Frontiers in psychology*, 4, 1015.

Sumner, M., & Samuel, A. G. (2009). The effect of experience on the perception and representation of dialect variants. *Journal of memory and language*, *60*(4), 487-501.

Walker, A., & Campbell-Kibler, K. (2015). Repeat what after whom? Exploring variable selectivity in a cross-dialectal shadowing task. Frontiers in Psychology, 6, 546.

Watson, C. I., Harrington, J., & Evans, Z. (1998). An acoustic comparison between New Zealand and Australian English vowels. *Australian journal of linguistics*, *18*(2), 185-207.

Wells, J. C. (1982). *Accents of English: Volume 1* (Vol. 1). Cambridge University Press.

Whorf, B. L. (1940). *Science and linguistics* (pp. 207-219). Indianapolis, IN, USA:: Bobbs-Merrill.

Witteman, M. J., Weber, A., & McQueen, J. M. (2014). Tolerance for inconsistency in foreign-accented speech. *Psychonomic bulletin & review*, *21*(2), 512-519.

Zheng, Y., & Samuel, A. G. (2017). Does seeing an Asian face make speech sound more accented? *Attention, Perception,& Psychophysics*, 79, 1841–1859.

Zimmerman, D. H. (1998). Identity, context and interaction. In C. Antaki & S. Widdicombe (Eds.), *Identities in talk* (pp. 87–106). Sage Publications Ltd.

Appendix 1: ExemPy code and documentation

```
"""
Created on Sat Sep 03 2022
@author: Emily Remirez (eremirez@berkeley.edu)

"""

"""Functions for implementing the Generalized Context Model for speech perception."""

import math
import random
import matplotlib.pyplot as plt
#%matplotlib inline
import numpy as np
import pandas as pd
from pandas import DataFrame
from scipy.optimize import minimize
import seaborn as sns

def activation(testset, cloud, dimsdict, c = 25):
    '''
    Calculate activation for all exemplars stored in the cloud
    with respect to some stimulus, referred to as test. Returns
    a data frame with column 'a' added for each row.

    Required parameters:

    testset = a dataframe with one or more rows, each a stimulus to be categorized
        must have columns matching those given in the 'dims' dict. These columns
        should be dimensions of the stimulus (e.g., formants)

    cloud = A dataframe of stored exemplars which every stimulus is compared to.
        Each row is an exemplar, which, like testset should have columns matching
        those in the dims dict

    dimsdict = a dictionary with dimensions as keys and weights, w, as values.

    c = an integer representing exemplar sensitivity. Defaults to 25.

    '''
    # Get stuff ready
    dims = dimsdict.copy()
    dims.update((x, (y/sum(dims.values()))) for x, y in dims.items())   # Normalize weights to sum to
1

    # If the testset happens to have N in it, remove it before joining dfs
    test = testset.copy()
    if 'N' in test.columns:
        test = test.drop(columns='N', axis=1,inplace=True)

    exemplars = cloud.copy()

    # Merge test and exemplars
    bigdf = pd.merge(
        test.assign(key = 1),         # Add column named 'key' with all values == 1
        exemplars.assign(key = 1),    # Add column named 'key' with all values == 1
        on = 'key',                   # Match on 'key' to get cross join (cartesian product)
        suffixes = ['_t', '_ex']
    ).drop('key', axis=1)           # Drop 'key' column


    dimensions = list(dims.keys())                # Get dimensions from dictionary
    weights = list(dims.values())                 # Get weights from dictionary
    tcols = [f'{d}_t' for d in dimensions]      # Get names of all test columns
    excols = [f'{d}_ex' for d in dimensions]    # Get names of all exemplar columns


    # Multiply each dimension by weights
    i = bigdf.loc[:, tcols].values.astype(float)     # Get all the test columns
    i *= weights                                     # Multiply test columns by weight
    j = bigdf.loc[:, excols].values.astype(float)    # Get all the exemplar columns
    j *= weights                                     # Multiply exemplar columns by weights

    # Get Euclidean distance
    bigdf['dist'] = np.sqrt(np.sum((i-j)**2, axis=1))

    # get activation: exponent of negative distance * sensitivity c, multiplied by N_j
    bigdf['a'] = np.exp(-bigdf.dist*c) * bigdf.N
    return bigdf


def exclude(cloud, test, exclude_self = True, alsoexclude = None):
    '''
    Removes specific rows from the cloud of exemplars, to be used
    prior to calculating activation. Prevents activation from being
    overpowered by stimuli that are too similar to particular exemplars.
    E.g., prevents comparison of a stimulus to itself, or to exemplars
    from same speaker. Returns dataframe containing a subset of
    rows from the cloud.

    Required parameters:

    cloud = A dataframe of stored exemplars which every stimulus is compared to.
        Each row is an exemplar

    test = single row dataframe containing the stimulus to be categorized

    exclude_self = boolean. If True, stimulus will be removed from exemplar cloud
        so that it isn't compared to itself. Defaults to True

    Optional parameters:

    alsoexclude = a list of strings matching columns in the cloud
        categories) to exclude if value is the same as that of the test.
        (E.g., to exclude all exemplars from
        the speaker to simulate categorization of novel speaker)
    '''
    # Make a copy of the cloud and call it exemplars.
    #    This is what we'll return at the end
    exemplars = cloud.copy()


    # Remove the stimulus from the cloud
    if exclude_self == True:
        exemplars = exemplars[~exemplars.isin(test)].dropna()

    if alsoexclude != None:
        if type(alsoexclude) != list:
            alsoexclude = [alsoexclude]
        for feature in alsoexclude:
            featval = test[feature].iloc[0]
            exclude_exemps = exemplars[exemplars[feature] == featval].index
            exemplars = exemplars.drop(exclude_exemps)

    return exemplars


def reset_N(exemplars, N = 1):
    '''
    Adds an N (base activation) column to the exemplar cloud so
    that activation with respect to the stimulus can be calculated
    Default value is 1, i.e., equal activation for each exemplar.
    Returns the exemplar data frame with added or reset column
    Required parameters:

    exemplars = data frame of exemplars to which the stimulus is being
        compared

    N = integer indicating the base activation value to be added to
        each exemplar (row) in the dataframe. Defaults to 1
    '''
    extemp = exemplars.copy()
    extemp['N'] = N
    return extemp


def bias_N(exemplars, cat, catbias):
    '''
    Adds or overwrites an N (base activation) columnn to the exemplar
    cloud so that activation with respect to the stimulus can be
    calculated. Unlike reset_N, which assigns the same N value to all exemplars,
    bias_N will set N values according to values in a dictionary.
    That is, within a category type, each category will have the N
    value specified in the dictionary

    Required parameters:

    exemplars = dataframe of exemplars to which the stimulus is being compared

    cat = a string designating the category type which is being primed

    catbias = dictionary with categories (e.g. vowels) as keys and N value for the
        category as values
    '''
    extemp = exemplars.copy()
    extemp['N'] = extemp[cat].map(catbias)
    return extemp


def probs(bigdf, cats):
    '''
    Calculates the probability that the stimulus will be categorized with a
    particular label for a given category (e.g., vowel labels 'i', 'a', 'u' for
    the category 'vowel'). Probability is calculated by summing the activation
    across all exemplars sharing a label, and dividing that by the total amount
    of activation in the system for the category. Returns a dictionary of dictionaries.
    Each key is a category; values are dictionaries where keys are labels and values
    represent probability of the stimulus being categorized into that label.

    Required parameters:

    bigdf = a dataframe produced by activation(), which contains a row for each
        exemplar with the additional column 'a' representing the amount of
        activation for that exemplar with respect to the stimulus

    cats = a list of strings containing at least one item, indicating which
        categories probability should be calculated for (e.g. ['vowel','gender']).
        Items should match the name of columns in the data frame
    '''
    prs = {}

    if type(cats) != list:
        cats = [cats]

    # Loop over every category in the list of categories
    for cat in cats:
        if cat in bigdf:
            label = cat
        else:
            # make category match the exemplar category in name if i and j share column names
            label = cat + '_ex'

        # Sum up activation for every label within that category
        cat_a = bigdf.groupby(label).a.sum()
        # Divide the activation for each label by the total activation for that category
        pr = cat_a / sum(cat_a)
        # rename a for activation to probability
        pr = pr.rename_axis(cat).reset_index().rename(columns={"a" : "probability"})
        # add this to the dictionary
        prs[cat] = pr
    return prs


def choose(probsdict, test, cats, fc = None):
    '''
    Chooses a label for each category which the stimulus will be categorized as.
    Returns the test/stimulus dataframe with added columns showing what was
    chosen for a category and with what probability. Optionally will give the
    second most probable label as well.

    Required parameters:
    pr = dictionary of probabilities, given from probs(). Each key should represent
        a category (e.g. 'vowel'), with values as dataframe. Dataframe should
        have a probability for each category label

    test = single line data frame representing the test/stimulus being categorized

    cats = list of categories to be considered (e.g., ["vowel"])

    Optional parameters:


    fc = Dict where keys are category names in the dataframe and
        values are a list of category labels.
        Used to simulate a forced choice experiment
        in which the perceiver has a limited number
        of alternatives. For example, if fc = {'vowel':['i','a']},
        the choice will be the alternative
        with higher probability, regardless of whether other vowels not
        listed have higher probabilities.
        There can be any number of alternatives in the list.

    '''
    newtest = test.copy()      # make a copy of the test set to add to
    pr = probsdict.copy()        # make a copy of the probs dict to subset if forced choice is set
    choice = ''
    choiceprob = 1

    # If using forced choice, restrict the choices to the terms
    # This doesn't change the probability! So something could have a low prob,
    ## but still be the winner
    if fc != None:
        fccats = fc.keys()
        for fccat in fccats:
            options = fc[fccat]
            scope = probsdict[fccat]
            toconsider = scope.loc[scope[fccat].isin(options)]
        pr[fccat] = toconsider

    for cat in cats:
        choicename = cat + 'Choice'
        choiceprobname = cat + 'Prob'

        dframe = pr[cat]
        prob = dframe['probability']
        winner = dframe.loc[prob==max(prob)]

        # if more than one winner, choose randomly
        if len(winner) > 1:
            winner = winner.sample(1)

        choice = winner[cat].item()
        choiceprob = winner['probability'].item()

        newtest[choicename] = choice
        newtest[choiceprobname] = choiceprob
    return newtest
```

```
def wideprobs(cats,pr):
    '''
    Get a list of probabilities reshaped to a wide format.

    Required parameters:

    cats = a list of strings containing at least one item, indicating which
        categories probability should be calculated for (e.g. ['vowel','gender']).
        Items should match the name of columns in the data frame

    pr = Output of probs function. Dictionary of dictionaries.
        Each key is a category; values are dictionaries where keys are
        labels and values represent probability of the stimulus
        being categorized into that label.
    '''
    widelist=[]
    for cat in cats:
        pref = str(cat+"_")
        dframe = pr[cat]
        wide = dframe.set_index(cat).transpose().reset_index(drop=True).add_prefix(pref)
        widelist.append(wide)
    return widelist

def probsdf(widelist, test):
    '''
    Alternative to the choose function. Rather than picking the category labels
    with the highest probability, join the wide format probalities alongside the stimulus.
    Returns the larger dataframe.

    Required parameters:

    widelist = Output of the wideprobs function. A list of probabilities for each category
        label, reshaped to wide format.

    test = single line data frame representing the test/stimulus being categorized
    '''
    newdf = test
    for wide in widelist:
        newdf = pd.merge(
            newdf.assign(key = 1),
            wide.assign(key = 1),
            on = 'key').drop('key', axis=1)
    return newdf


def categorize(testset, cloud, cats, dimsdict, c,
               exclude_self = True, alsoexclude = None, N=1, fc=None):
    '''
    Categorizes a stimulus based on functions defined in library.
    1. Exclude any desired stimuli
    2. Add N value
    3. Calculate activation
    4. Calculate probabilities
    5. Choose labels for each category
    Returns the output of choose(): test/stimulus dataframe with
    added columns showing what was
    chosen for a category and with what probability

    Required parameters:

    testset = a dataframe with one row, a stimulus to be categorized
        must have columns matching those given in the 'dims' dict. These columns
        should be dimensions of the stimulus (e.g., formants)

    cloud = A dataframe of stored exemplars which every stimulus is compared to.
        Each row is an exemplar, which, like testset should have columns matching
        those in the dims dict

    cats = a list of strings containing at least one item, indicating which
        categories probability should be calculated for (e.g. ['vowel','gender']).
        Items should match the name of columns in the data frame
    dimsdict = a dictionary with dimensions as keys and weights, w, as values.

    c = an integer representing exemplar sensitivity. Defaults to .01.

    exclude_self = boolean. If True, stimulus will be removed from exemplar cloud
        so that it isn't compared to itself. Defaults to True

    Optional parameters:
    alsoexclude = a list of strings matching columns in the cloud (categories)
        to exclude  if value is the same as that of the test.
        (E.g., to exclude all exemplars from the speaker
        to simulate categorization of novel speaker)

    N = integer indicating the base activation value to be added to
        each exemplar (row) in the dataframe. Defaults to 1



    '''
    exemplars = cloud.copy()
    test = testset
    exemplars = exclude(exemplars, test, exclude_self = exclude_self, alsoexclude = alsoexclude)
    exemplars = reset_N(exemplars, N = N)
    bigdf = activation(test, exemplars, dimsdict = dimsdict, c = c)
    pr = probs(bigdf, cats)
    choices = choose(pr, test, cats, fc = fc)
    return choices


def multicat(testset, cloud, cats, dimsdict, c = 25, N = 1, biascat = None, catbias = None,
                 exclude_self = True, alsoexclude = None,  fc = None):
    '''
    Categorizes a dataframe of 1 or more stimuli based on functions defined in library

    1. Exclude any desired stimuli
    2. Add N value
    3. Calculate activation
    4. Calculate probabilities
    5. Choose labels for each category
    Returns the output of choose(): test/stimulus dataframe with added columns
    showing what was chosen for a category and with what probability

    Required parameters:

    testset = a dataframe with one or more rows, each a stimulus to be categorized
        must have columns matching those given in the 'dims' dict. These columns
        should be dimensions of the stimulus (e.g., formants)

    cloud = A dataframe of stored exemplars which every stimulus is compared to.
        Each row is an exemplar, which, like testset should have columns matching
        those in the dims dict

    cats = a list of strings containing at least one item, indicating which
        categories probability should be calculated for (e.g. ['vowel','gender']).
        Items should match the name of columns in the data frame

    dimsdict = a dictionary with dimensions as keys and weights, w, as values.

    c = an integer representing exemplar sensitivity. Defaults to 25.

    exclude_self = boolean. If True, stimulus will be removed from exemplar cloud
        so that it isn't compared to itself. Defaults to True

    Optional parameters:

    biascat = A string indicating the category type to be biased
        or primed on (e.g. 'vowel', 'speaker')

    catbias = Dict where keys are categories of biascat and values are
        ints that indicate relative N values. (e.g., {'i':5,'a':1} would make every 'i' exemplar
        contribute 5 times as much activation as each 'a)
    alsoexclude = a list of strings matching columns in the cloud (categories) to exclude
        if value is the same as that of the test. (E.g., to exclude all exemplars from
        the speaker to simulate categorization of novel speaker)

    N = integer indicating the base activation value to be added to
        each exemplar (row) in the dataframe. Defaults to 1


    fc = Dict where keys are category names in the dataframe and values are a list of category
labels.
        Used to simulate a forced choice experiment in which the perceiver has a limited number
        of alternatives. For example, if fc = {'vowel':['i','a']}, the choice will be the alternative
        with higher probability, regardless of whether other vowels not listed have higher
probabilities.
        There can be any number of alternatives in the list.


    '''
    choicelist=[]
    prlist=[]
    for ix in list(testset.index.values):
        # Reload exemplars within the loop
        ## if not, exemplars shrinks every time you use exclude()!
        exemplars = cloud.copy()
        test = testset.loc[[ix,]]

        # exclusions
        exemplars = exclude(exemplars, test, exclude_self = exclude_self, alsoexclude = alsoexclude)

        #add N
        if catbias != None:
            exemplars = bias_N(exemplars, biascat, catbias)
        else: exemplars = reset_N(exemplars, N = N)

        # calculate probabilities
        bigdf = activation(test, exemplars, dimsdict = dimsdict, c = c)
        pr = probs(bigdf, cats)

        # Luce's choice rule
        choicerow = choose(pr, test, cats, fc = fc)
        choicelist.append(choicerow)
        # Get probabilities
        widelist = wideprobs(cats, pr)
        widerow = probsdf(widelist,test)
        widelist.append(widerow)
    choices = pd.concat(choicelist, ignore_index = True)

    return choices
```

```
"""
Created on Sat Sep 03 2022
@author: Emily Remirez (eremirez@berkeley.edu)

"""

"""Utility functions for implementing and evaluating the Generalized Context Model for speech
perception."""

import math
import random
import matplotlib.pyplot as plt
#%matplotlib inline
import numpy as np
import pandas as pd
from pandas import DataFrame
from scipy.optimize import minimize
import seaborn as sns

def HzToBark(cloud, formants):
    '''
    Convert selected columns from Hz to Bark scale. Renames the formants as z.
    Returns the data frame with additional columns: the value of the formant
    converted from Hz to Bark

    Required parameters:

    cloud = dataframe of exemplars

    formants = list of formants to be converted
    '''
    # Make a copy of the cloud
    newcloud = cloud.copy()

    # For each formant listed, make a copy of the column prefixed with z
    for formant in formants:
        for ch in formant:
            if ch.isnumeric():
                num = ch
        formantchar = (formant.split(num)[0])
        name = str(formant).replace(formantchar, 'z')
        # Convert each value from Hz to Bark
        newcloud[name] = 26.81 / (1 + 1960 / newcloud[formant]) - 0.53
    return newcloud

def gettestset(cloud, balcat, n):     #Gets n number of rows per cat in given cattype
    '''
    Gets a random test set of stimuli to be categorized balanced across a particular
    category, e.g., 5 instances of each label 'i','a', 'u' for category 'vowel'.
    Returns a data frame of stimuli.

    Required parameters:

    cloud = dataframe of exemplars

    balcat = category stimuli should be balanced across

    n = number of stimuli per category label to be included
    '''
    testlist=[]
    for cat in list(cloud[balcat].unique()):
        samp = cloud[cloud[balcat] == cat].sample(n)
        testlist.append(samp)
    test = pd.concat(testlist)
    return test


def checkaccuracy(choices, cats):
    '''
    Check rather the choices made by the model match the 'intended' label for each category.
    Returns a copy of the testset dataframe with column added indicating whether the choice for
    each category was correct (y) or incorrect (n)
    Required parameters:

    choices = output of choose() function: the test/stimulus dataframe with added columns showing
what was
        chosen for a category and with what probability.

    cats = a list of strings containing at least one item, indicating which
        category's probability was calculated for (e.g. ['vowel','gender']).
        Items should match the name of columns in the data frame
    '''
    if type(cats) != list:
        cats = [cats]

    acc = choices.copy()
    for cat in cats:                      # Iterate over your list of cats
        accname = cat + 'Acc'             # Get the right column names
        choicename = cat + 'Choice'

        # If choice is the same as intended, acc =y, else n
        acc[accname] = np.where(acc[cat]==acc[choicename], 'y', 'n')
    return acc


def propcorr(acc, cat):
    '''
    Calculates the proportion of stimuli under each label which were categorized correctly
    Returns a dataframe with keys as labels and values as proportions between 0 and 1.

    Required parameters:

    acc = output of checkaccuracy() function: a copy of the testset dataframe with column
        added indicating whether the choice for each category was correct (y) or incorrect (n)

    cat = string ndicating which category accuracy should be assessed for. String should match
        column in acc.
    '''
    perc = dict(acc.groupby(cat)[cat+'Acc'].value_counts(normalize = True).drop(labels = 'n',level =
1).reset_index(level = 1,drop = True))
    pc = pd.DataFrame.from_dict(perc, orient = 'index').reset_index()
    pc.columns = [cat,'propcorr']
    return pc


def overallacc(acc, cat):
    '''
    Calculates accuracy for categorization overall, across all labels. Returns a
    proportion between 0 and 1.

    Required parameters:

    acc = output of checkaccuracy() function: a copy of the testset dataframe with column
        added indicating whether the choice for each category was correct (y) or incorrect (n)

    cat = string ndicating which category accuracy should be assessed for. String should match
        column in acc.
    '''

    totalcorrect = acc[cat+'Acc'].value_counts(normalize=True)['y']
    return totalcorrect

def confusion(choices, cats):
    '''
    Returns a confusion matrix comparing intended category with categorization.

    Required parameters:

    choices = output of choose() function: the test/stimulus dataframe with added columns showing
what was
        chosen for a category and with what probability.

    cats = a list of strings containing at least one item, indicating which
```

```
        categories probability was calculated for (e.g. ['vowel','gender']).
        Items should match the name of columns in the data frame
    '''
    if type(cats) != list:
        cats = [cats]

    matrices = {}
    for cat in cats:
        matrices[cat] = pd.crosstab(choices[cat],
                                    choices[cat+'Choice'],
                                    normalize ='index').round(2).rename_axis(None)
    return matrices


def errorfunc(x, testset, cloud, dimslist, cat):
    '''
    Returns a proportion representing the total amount of error for a single category that
    the categorizer makes given a certain set of c and w values. This is intended to
    be used with an optimization function so that the total amount of error can be
    minimized; that is, the accuracy can be maximized.
    Note that z0 is automatically set to 1.

    Required parameters:

    x = a vector of values to be used by multicat. x[0] should be c, x[1], x[2], x[3]
        should correspond to dimslist[1], dimslist[2], dimslist[3]

    testset = a dataframe with one or more rows, each a stimulus to be categorized
        must have columns matching those given in the dims list. These columns
        should be dimensions of the stimulus (e.g., formants)

    cloud = A dataframe of stored exemplars which every stimulus is compared to.
        Each row is an exemplar, which, like testset should have columns matching
        those in the dims list

    dimslist = a list of dimensions (e.g., formants), for which weights w should be given,
        and along which exemplars should be compared.

    cat = the category,


    '''
    #x = [c,z1,z2,z3]
    catlist = [cat]
    c = x[0]
    dimsdict = {dimslist[0] : 1,
                dimslist[1] : x[1],
                dimslist[2] : x[2],
                dimslist[3] : x[3]}
    choices = multicat(cloud, testset, catlist, dims = dimsdict, c = c)
    accuracy = checkaccuracy(choices, catlist)
    err = accuracy[cat+'Acc'].value_counts(normalize = True)['n']
    return err


def continuum (data, start, end, dimslist, steps = 7, stimdetails = False, addcol=None):
    '''
    Returns a continuum dataframe with interpolated values
    from a start to end value with a given number of steps
    * Users should be sure to specify any and all parameters they want
    start and end to match for. That is, say there are 2 repetitions of
    a stimulus. If it doesn't matter whether start and end are from the same
    repetition, you do not need to specify repetition number; one row will
    be chosen randomly. If it *does* matter that they're the same repetition,
    be sure to include repetition number in the dictionary.

    Required parameters:

    data = DataFrame to draw start and end stimuli from

    start = Dictionary indicating properties of the desired start
        with category types as keys, and their desired category as values.
        e.g., {"vowel":"i","speaker"="LB"}

    end = Dictionary indicating properties of the desired start
        with category types as keys, and their desired category as values

    dimslist = list containing the names of dimensions to be interpolated

    Optional parameters:

    steps = integer indicating the total number of continuum steps. Defaults to 7.

    stimdetails = Boolean, defaults to False. Debugging/auditing tool to
        get details of the stimulus that aren't preserved in the returned
        dataframe (e.g., speaker ID)

    addcols = Dictionary indicating a column to preserve in the continuum, e.g., s
    '''
    # create a copy of the entire df to subset according to conditions
    # match category to value from dictionary, subset
    # repeat subsetting until all conditions are satisfied
    st = data.copy()
    for i in range(0, len(start)):
        cat = list(start.keys())[i]
        val = list(start.values())[i]
        condition = st[cat] == val
        st = st.loc[condition]
    # reset index has to be outside of the loop to work with >2 conditions
    # sample(1) is there to just pick an observation if the conditions don't point
    ## a unique row in the dataframe
    st = st.sample(1).reset_index()

    en = data.copy()
    for i in range(0,len(end)):
        cat = list(end.keys())[i]
        val = list(end.values())[i]
        condition = en[cat] == val
        en = pd.DataFrame(en.loc[condition])
    en = en.sample(1).reset_index()

    # remember start & end values if needed
    if stimdetails == True:
        print("Start: " , st.iloc[0])
        print("End: " , en.iloc[0])

    norms = {}
    for dim in dimslist:                      # Calculate the difference between start and end for
each dim
        norms[dim] = en[dim] - st[dim]


    rowlist = []
    for i in range (0, steps):
        vals = {"step" : (i+1)}
        if addcol != None:
            vals.update(addcol)
        for dim in dimslist:
            vals[dim] = st[dim] + (norms[dim] * i/(steps-1))    # the values for each dim = start val
+ diff by step
            row = pd.DataFrame(vals)
        rowlist.append(row)

    contdf = pd.concat(rowlist, ignore_index = True)
    return contdf


def datasummary(dataset, catslist, dimslist):
    '''
    Creates dataframe of mean values grouped by catgories

    Required parameters:

    dataset = A dataframe to be analyzed, where each row is an observation
        Requires at least one category and one dimension

    catslist = List of categories to group by. Also accepts string.

    dimslist = List of dimensions to get values for. Also accepts dict
        with dimensions as keys.
    '''
    # Convert cat to list (e.g. if only one term is given)
    if type(catslist) != list:
        catslist = [catslist]
    # If the weights dictionary is given instead of the dimlist,
    ## take just the keys as a list
    if type(dimslist) == dict:
        dimslist = list(dimslist.keys())

    # group by categories: cats[0] will be used to group first, then cats[1]
    # i.e., if cats = ["vowel","type"], vowel1-type1, vowel1-type2, vowel2-type1, vowel2-type2...
    # get the mean of values for each dimension grouped by categories
    d = dataset.copy()
    df = d.groupby(catslist, as_index = False)[dimslist].mean()
    return df
```

```
"""
Created on Sat Sep 03 2022
@author: Emily Remirez (eremirez@berkeley.edu)

"""

"""Functions for visualizing ExemPy simulations."""

import math
import random
import matplotlib.pyplot as plt
#%matplotlib inline
import numpy as np
import pandas as pd
from pandas import DataFrame
from scipy.optimize import minimize
import seaborn as sns

def getactiv(activxn, x, y, cat):

    """
    Creates a simplified data frame showing the activation for each exemplar
    with respect to the stimulus. Primarily for use with the activplot()
    function.

    Required parameters:

    actxn = DataFrame resulting from the activation() function, containing
        one row per stored exemplar, with their activation 'a' as a column

    x = String. Dimension to be plotted as x axis in scatterplot (e.g., F2). Matches
        the name of a column in the activation DataFrame.

    y = String. Dimension to be plotted as y axis in scatterplot (e.g., F1). Matches
        the name of a column in the activation DataFrame.

    cat = String. Category used to color code exemplars in scatter plot. Matches the name
        of a column in the activation DataFrame.
    """

    renamedict = {}
    activseries={"a" : activxn['a']}

    for item in (x, y, cat):
        name = str(item + "_ex")
        if name not in activxn:
            name = item
        renamedict[name] = item
        activseries[item] = activxn[name]
    activ = pd.DataFrame.from_dict(activseries)
    return activ



def activplot(a, x, y, cat, test, invert = True, catlegend = False):
    """
    Plots each exemplar in x,y space according to specified dimensions. Labels within
    the category are grouped by color. The stimulus or test exemplar is plotted in dark
    blue on top of exemplars. Note: axes are inverted, assuming F1/F2 space

    Required parameters:

    a = DataFrame produced by getactiv() function. Contains a row for each exemplar

    x = String. Dimension to be plotted as x axis in scatterplot (e.g., F2). Matches
        the name of a column in the activation DataFrame.

    y = String. Dimension to be plotted as y axis in scatterplot (e.g., F1). Matches
        the name of a column in the activation DataFrame.

    cat = String. Category used to color code exemplars in scatter plot. Matches the name
        of a column in the activation DataFrame.
    test = name of test exemplar, one row of a DataFrame.

    Optional parameters:

    invert = Boolean. Specifies whether axes should be inverted (as for a vowel space). Defaults to
true.

    catlegend = Boolean. Specifies whether to show a legend for the category. Defaults to false.

    """


    pl = sns.scatterplot(data = a,
                         x = x,
                         y = y,
                         hue = cat,
                         size = 'a',
                         size_norm = (0, a.a.max()),
                         alpha = 0.5,
                         sizes = (5, 100),
                         legend = catlegend)


    pl = sns.scatterplot(data = test,
                         x = x,
                         y = y,
                         alpha = .5,
                         color = 'darkblue',
                         marker = "X",
                         s = 50,
                         legend = False)


    if invert == True:
        pl.invert_xaxis()
        pl.invert_yaxis()
    return pl


def accplot(acc, cat, **kwargs):
    '''
    Plots a bar graph showing the proportion of trials which were categorized
    veridically, that is, accuracy of categorization.

    Required parameters:

    acc = output of checkaccuracy() function: a copy of the testset dataframe with column
        added indicating whether the choice for each category was correct (y) or incorrect (n)

    cat = string ndicating which category accuracy should be assessed for. String should match
        column in acc.

    '''
    perc = dict(
        acc.groupby(cat)[cat+'Acc']
            .value_counts(normalize = True)
            .drop(labels = 'n', level = 1)
            .reset_index(level = 1, drop = True))
    pc = pd.DataFrame.from_dict(perc, orient = 'index').reset_index()
    pc.columns = [cat,'propcorr']

    obs = str(len(acc))
    pl = sns.barplot(x = cat, y = 'propcorr', data = pc)
    plt.ylim(0,1.01)
    pl.set(ylabel = 'Proportion accurate of ' + obs + ' trials')
    pl.set_xticklabels(
        pl.get_xticklabels(),
        rotation = 45,
        horizontalalignment = 'right',
        fontweight = 'light',
        fontsize = 'x-large')
    plt.show()
    return pl

def cpplot(datalist, cat, xax, datanames = None, plot50 = True, stv=None, env=None):
    '''
    Generates a (cp = categorical perception) plot. On the X axis is the stimulus number,
    on the Y axis is the proportion of [label] responses with [label] being the label that
    was assigned to the first stimulus. Designed to be used with stimuli continua

    Required parameters:

    datalist = Designed to be output of multicat() or multicatprime(). Dataframe or list of
dataframes
        containing each stimulus, what it was categorized as, and the probability

    cat = Type of category decision to visualize, e.g., 'vowel'

    Optional parameters:

    xax = String giving the name of a column to use as the x-axis on the plot; for example, the step
        number along a continuum

    datanames = List of labels to use for each curve in the plot. Names should be in same
        order as in datalist

    plot50 = Boolean indicating whether a dashed line is added at 0.5 to aid in assessing
        boundaries in categorical perception. Defaults to true.
    '''
    if type(datalist) != list:
        datalist = [datalist]
    choicename = cat + 'Choice'
    probname = cat + 'Prob'

    if stv == None:
        stv = datalist[0].loc[0][choicename] #start value
    if env == None:
        env = datalist[0].iloc[-1][choicename] #end value

    def copy(d):
        d = d
        return d
    def inv(d):
        d = 1-d
        return d

    j = 0
    for dataset in datalist:
        if datanames != None:
            lab = datanames[j]
        else:
            lab = "Data " + str(i)
        ## get the inverse of probability if not first value, for each dataset
            # If neither start nor end, set value to NaN
        dataset['yax'] = dataset.apply(
            lambda x: copy(x[probname]) if x[choicename] == stv
            else (inv(x[probname]) if x[choicename] == env else float("NaN")),
            axis = 1)
        dataset["Data"] = lab
        j += 1

    datalist = pd.concat(datalist)
    curve = sns.pointplot(
            x = xax,
            y = "yax",
            data = datalist,
            hue = "Data")

    for line, row in datalist.iterrows():
        if math.isnan(row["yax"]):
            realchoice = str(row[choicename])
            realprob = str(np.round(row[probname], 2))
            stimulusnumber = str(row[xax])
            dataset = str(row["Data"])
            print("----- Hey! -----")
            print("Stimulus", stimulusnumber)
            print("in dataset", dataset)
            print("was categorized as", realchoice)
            print("with probability", realprob)

    # Add labels & plot
    yaxisname = "Proportion " + stv + " Response"
    curve.set_ylabel(yaxisname)
    curve.set_xlabel("Step")
    curve.set_ylim(-0.05, 1.05)

    if plot50 == True:
        plt.axhline(y = 0.5, color = 'gray', linestyle = ':')

    return curve
```

Appendix 2: ExemPy demonstrations

Basic functions

In [1]:

```
%load_ext autoreload
from ExemPy import *
from ExemPy.utils import *
from ExemPy.viz import *
from ExemPy.GCM import *
%aimport ExemPy, ExemPy.utils, ExemPy.viz, ExemPy.GCM
%autoreload 1
import math
import random
import matplotlib.pyplot as plt
#%matplotlib inline
import numpy as np
import pandas as pd
from pandas import DataFrame
from scipy.optimize import minimize
import seaborn as sns
sns.set(style='ticks', context='paper')
colors=["#e3c934","#68c4bf","#c51000","#287271"]
sns.set_palette(colors)
```

Set up data

- Read in Peterson and Barney 1952
- Convert Hz to Bark
- Read in confusion matrix from PB52
- Preview pb52

In [2]:

```
pb52 = pd.read_csv('pb52_data//pb52.csv')
pbcm = pd.read_csv('pb52_data//pbcm.csv').drop([0]).set_index
hgcm = pd.read_csv('pb52_data//hgcm.csv').drop([0]).set_index
pb52 = HzToBark(pb52, ["F0", "F1", "F2", "F3"])
pb52.sample(5)
```

Out[2]:

	type	gender	speaker	vowel	repetition	F0	F1	F2
1161	w	f	59	FLEECE	2	261	280	2740
1383	c	m	70	KIT	2	212	420	2480
1197	w	f	60	GOOSE	2	188	340	920
1492	c	m	75	THOUGHT	1	270	535	970
981	w	f	50	FLEECE	2	200	400	2600

Set default parameters (based on previous work)

- dimensions *m* & attention weights *w*
- list of categories
- exemplar sensitivity *c*
- set to be categorized
- exemplar cloud used to categorize against

In [3]:

```
dimsvals={'z0' : 1,
          'z1' : 2.56,
          'z2' : 1.985,
          'z3' : 1.34}

dimslist = list(dimsvals.keys())

catslist = ['type', 'vowel']            # man, woman, or child

cval = 25
```

Categorize one stimulus

You can also use the function xm.categorize() to do these steps all at once! (Note, however, that you won't be able to visualize and verify at each step of the process!)

Set up

- Randomly select a stimulus

- Add an resting activation *N* value of 1to all stored exemplars
- Exclude the stimulus from the exemplar cloud
  - This way, the stimulus will not be compared to itself

In [4]:

```
stim = pb52.sample()
stim
```

Out[4]:

	type	gender	speaker	vowel	repetition	F0	F1	F2	F3
855	w	f	43	FOOT	2	233	467	1167	2595

In [5]:

```
exemplars = reset_N(pb52, N = 1)
exemplars = exclude(exemplars, stim, exclude_self = True)
print(exemplars.sample())
```

```
    type gender  speaker  vowel  repetition     F0     F1
F2      F3  \
685    w      f     35.0  DRESS         2.0  210.0  630.0  230
0.0  3170.0

           z0        z1          z2          z3  N
685  2.064516  5.991351  13.944883  16.036803  1
```

In [6]:

```
o = stim.isin(exemplars)
print('Is stim contained within exemplars?')
print(o)
```

```
Is stim contained within exemplars?
       type  gender  speaker  vowel  repetition     F0     F1
F2     F3  \
855  False   False    False  False       False  False  False
False  False

        z0     z1     z2     z3
855  False  False  False  False
```

Calculate activation of each exemplar wrt the stimulus

- For exemplars that share categories with stimulus,
  - activation (a) *a* is higher
  - distance (dist) *d* is lower
- Plot activation:
  - exemplars plotted in F2, F1 (Bark) space
  - stimulus plotted in a blue X

- color = exemplar category
- size = activation

In [7]:

```
activation_df = activation(
    testset = stim,
    cloud = reset_N(exemplars, N=1),
    dimsdict = dimsvals,
    c = cval
    )
```

In [8]:

```
print("---- Stimulus info ----")
print(stim[['type', 'vowel', 'speaker']])
print("")
print(stim[['F0', 'F1', 'F2', 'F3']])
print("-----------------------")
smallactivdf = activation_df[['a',
                      'dist',
                      'type_ex',
                      'vowel_ex',
                      'F0_ex',
                      "F1_ex",
                      "F2_ex",
                      "F3_ex"]]
print(gettestset(smallactivdf, "vowel_ex", n = 2))
```

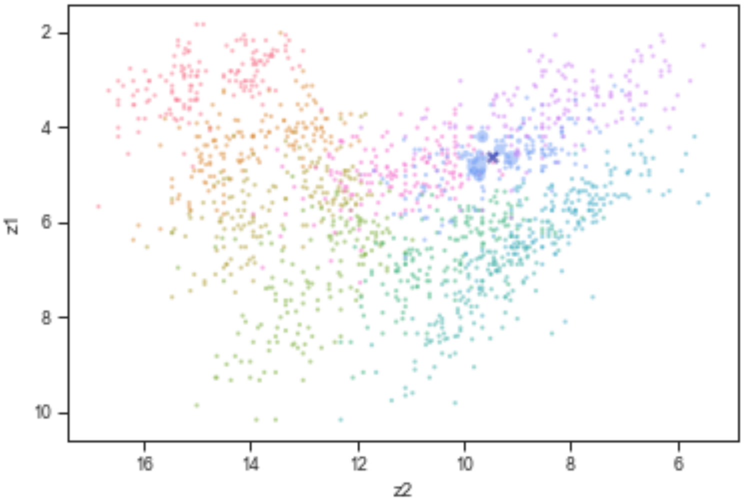
```
---- Stimulus info ----
    type vowel  speaker
855    w  FOOT       43

      F0   F1    F2    F3
855  233  467  1167  2595
-----------------------
                 a      dist type_ex vowel_ex  F0_ex   F1_ex
F2_ex   F3_ex
160   5.916923e-25  2.231472       m   FLEECE  175.0   316.0
2200.0  2800.0
700   7.340776e-33  2.959675       w   FLEECE  210.0   290.0
2700.0  3020.0
1261  4.186318e-34  3.074243       c      KIT  290.0   580.0
2760.0  3400.0
483   2.311896e-19  1.716442       m      KIT  108.0   430.0
1940.0  2590.0
924   6.340351e-27  2.412915       w    DRESS  260.0   520.0
2340.0  3040.0
1203  1.499031e-21  1.917979       w    DRESS  245.0   640.0
1980.0  2920.0
1006  2.697548e-29  2.631305       w     TRAP  203.0   678.0
2420.0  3080.0
267   2.640673e-18  1.619020       m     TRAP  109.0   750.0
1710.0  2440.0
1008  2.242467e-12  1.072938       w    STRUT  214.0   772.0
1280.0  2660.0
449   9.100084e-10  0.832703       m    STRUT  124.0   650.0
1000.0  2520.0
50    1.479473e-11  0.997470       m     PALM   91.0   640.0
1080.0  2140.0
1249  3.393938e-27  2.437912       c     PALM  265.0  1170.0
1500.0  3440.0
1411  5.237315e-15  1.315319       c  THOUGHT  232.0   670.0
1160.0  3550.0
1111  2.266704e-10  0.888301       w  THOUGHT  229.0   688.0
1029.0  2750.0
135   5.752845e-06  0.482633       m     FOOT  170.0   493.0
1120.0  2300.0
215   7.663624e-08  0.655368       m     FOOT  143.0   430.0
1030.0  2275.0
536   3.500509e-15  1.331435       m    GOOSE  133.0   294.0
930.0  2050.0
1036  4.321375e-05  0.401974       w    GOOSE  188.0   375.0
1143.0  2700.0
479   5.157253e-18  1.592245       m    NURSE  150.0   600.0
1470.0  1820.0
1097  1.206244e-16  1.466154       w    NURSE  213.0   533.0
```

```
        1425.0  1830.0
```

```
In [9]: act = getactiv(activation_df, 'z2', 'z1', 'vowel')
```

```
In [10]: activplot(act, 'z2', 'z1', 'vowel', stim)
```

```
Out[10]: <AxesSubplot:xlabel='z2', ylabel='z1'>
```



Calculate probabilities & choose

Use Luce's choice rule to calculate probabilities & choose

- Calculate probability for categorization as each label
  - Add up activation by category
  - Add up total amount of activation
  - Divide category activation by total activation
    - Probability of being categorized as that
- Choose the label with the highest probability

```
In [11]: pr = probs(activation_df, catslist)
         pr
```

Out[11]:

```
{'type':   type  probability
 0    c     0.041233
 1    m     0.110480
 2    w     0.848287,
 'vowel':       vowel   probability
 0    DRESS  1.011615e-12
 1   FLEECE  2.450425e-21
 2     FOOT  9.522804e-01
 3    GOOSE  4.659325e-02
 4      KIT  7.843954e-14
 5    NURSE  1.351160e-11
 6     PALM  1.100953e-04
 7    STRUT  8.571126e-04
 8  THOUGHT  1.591088e-04
 9     TRAP  1.147804e-10}
```

In [12]:

```
choices = choose(pr, stim, catslist)
choices[['typeChoice','typeProb','vowelChoice','vowelProb']]
```

Out[12]:

	typeChoice	typeProb	vowelChoice	vowelProb
855	w	0.848287	FOOT	0.95228

Check for "veridical perception"

In [13]:

```
accu = checkaccuracy(choices, catslist)
print('Was vowel categorized accurately?          '
      + accu.iloc[0]['vowelAcc'])
print('Was speaker type categorized accurately?   '
      + accu.iloc[0]['typeAcc'])
```

```
Was vowel categorized accurately?          y
Was speaker type categorized accurately?   y
```

Categorize dataset with respect to itself

In the past sections we:

- Set some parameters for categorization, including
  - Dimensions & their attention weights
  - Category types to categorize for
- Calculate the activation of each exemplar wrt the stimulus
- Calculate probabilities
- Choose the most probable category label for each category type

- Evaluate the stimulus

Now, let's do the same process with **multiple** exemplars. In fact, let's simulate the perception component of Peterson and Barney 1952 by having our "perceiver" categorize every exemplar.

First, let's re-set our variables

As a reminder or in case something got wonky!

In [14]:

```
dimsvals={'z0' : 1,
          'z1' : 2.56,
          'z2' : 1.985,
          'z3' : 1.34}

dimslist = list(dimsvals.keys())

catslist = ['type', 'vowel']

cval = 25

exemplars = pb52
test = pb52
```

Next, let's categorize!

This may take a minute!

In [15]:

```
choices = multicat(
    testset = test,
    cloud = exemplars,
    cats = catslist,
    dimsdict = dimsvals,
    c = cval,
    N = 1,
    exclude_self = True)
```

In [16]:

```
print(choices)
```

```
     type gender  speaker   vowel  repetition   F0   F1    F2
F3        z0  \
0       m      m        1  FLEECE           1  160  240  2280
2850  1.493396
1       m      m        1  FLEECE           2  186  280  2400
2790  1.793700
2       m      m        1     KIT           1  203  390  2030
2640  1.986149
3       m      m        1     KIT           2  192  310  1980
2550  1.861970
4       m      m        1   DRESS           1  161  490  1870
2420  1.505083
...   ...    ...      ...     ...         ...  ...  ...   ...
...       ...
1515    c      f       76    FOOT           2  322  610  1550
3400  3.253006
1516    c      f       76   GOOSE           1  345  520  1250
3460  3.482777
1517    c      f       76   GOOSE           2  334  500  1140
3380  3.373461
1518    c      f       76   NURSE           1  308  740  1850
2160  3.110864
1519    c      f       76   NURSE           2  328  660  1830
2200  3.313392

            z1          z2          z3 typeChoice  typeProb vowe
lChoice  \
0     2.394727  13.886698  15.355343          m  0.999809
FLEECE
1     2.821250  14.227798  15.217347          m  0.994734
FLEECE
2     3.919319  13.110175  14.856609          m  0.982181
KIT
3     3.131278  12.943046  14.628647          m  0.999517
KIT
4     4.832000  12.560000  14.282831          m  0.739049
DRESS
...         ...         ...         ...         ...       ...
...
1515  5.833463  11.309174  16.476343          c  0.999919
FOOT
1516  5.091452   9.910031  16.584871          c  0.993950
FOOT
1517  4.919187   9.329161  16.439625          c  0.998422
FOOT
1518  6.817926  12.487979  13.525728          c  0.495870
NURSE
1519  6.223664  12.415198  13.648365          c  0.855784
```

```
NURSE

      vowelProb
0      0.999925
1      0.998462
2      0.973843
3      0.996075
4      0.572512
...         ...
1515   0.983871
1516   0.934974
1517   0.825487
1518   0.503462
1519   0.898766

[1520 rows x 17 columns]
```

Check accuracy

- We'll get a dataframe where each exemplar is labeled with whether or not it was categorized accurately ([CATNAME]Acc)

Then, for each category type:

- We'll tabulate and plot accuracy by category label
- And then look at overall accuracy as a proportion

In [17]:

```
acc = checkaccuracy(choices,catslist)
acc.sample(5)
```

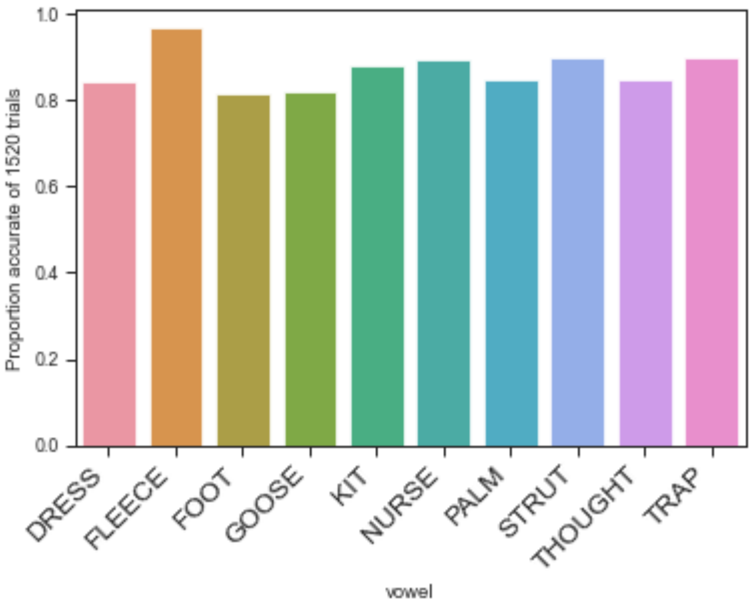
Out[17]:

	type	gender	speaker	vowel	repetition	F0	F1	F2	
981	w	f	50	FLEECE	2	200	400	2600	3
345	m	m	18	DRESS	2	129	490	1930	2
1198	w	f	60	NURSE	1	222	530	1670	2
1247	c	f	63	TRAP	2	285	1140	2000	3
1189	w	f	60	STRUT	2	214	770	1530	2

For vowel

In [18]:

```
accplot(acc, 'vowel')
```



```
Out[18]: <AxesSubplot:xlabel='vowel', ylabel='Proportion accurate of
         1520 trials'>
```

```
In [19]: propcorr(acc,'vowel')
```

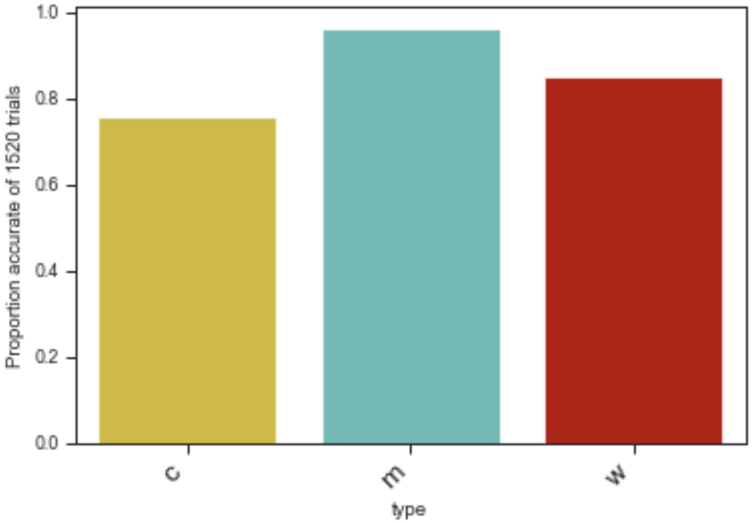
Out[19]:

	vowel	propcorr
0	DRESS	0.842105
1	FLEECE	0.967105
2	FOOT	0.815789
3	GOOSE	0.822368
4	KIT	0.881579
5	NURSE	0.894737
6	PALM	0.848684
7	STRUT	0.901316
8	THOUGHT	0.848684
9	TRAP	0.901316

```
In [20]: print("Overall accuracy: " + str(overallacc(acc,'vowel')))

        Overall accuracy: 0.8723684210526316
```

For speaker type (man, woman, child)

```
In [21]: accplot(acc,'type')
```



```
Out[21]: <AxesSubplot:xlabel='type', ylabel='Proportion accurate of 1
         520 trials'>
```

```
In [22]: propcorr(acc,'type')
```

Out[22]:

	type	propcorr
0	c	0.756667
1	m	0.962121
2	w	0.851786

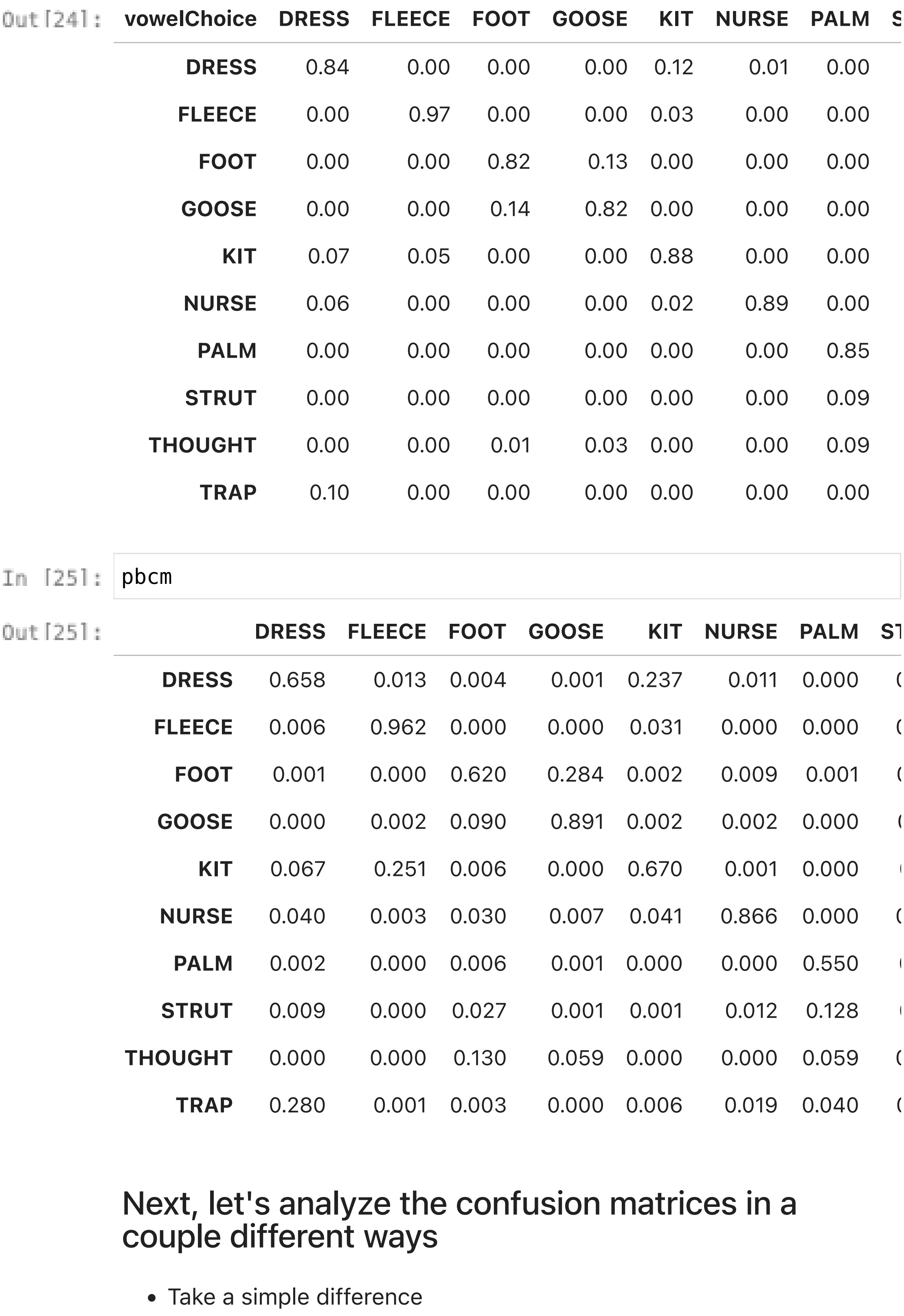
```
In [23]: print("Overall accuracy: " + str(overallacc(acc,'type')))
```

```
Overall accuracy: 0.8809210526315789
```

Get confusion matrices

- Calculate a confusion matrix for our "perceiver"
- Look at the confusion matrix from the Peterson and Barney paper we read in earlier

```
In [24]: cm = confusion(choices,catslist)['vowel']
         cm
```



Out[24]:

vowelChoice	DRESS	FLEECE	FOOT	GOOSE	KIT	NURSE	PALM	S
DRESS	0.84	0.00	0.00	0.00	0.12	0.01	0.00	
FLEECE	0.00	0.97	0.00	0.00	0.03	0.00	0.00	
FOOT	0.00	0.00	0.82	0.13	0.00	0.00	0.00	
GOOSE	0.00	0.00	0.14	0.82	0.00	0.00	0.00	
KIT	0.07	0.05	0.00	0.00	0.88	0.00	0.00	
NURSE	0.06	0.00	0.00	0.00	0.02	0.89	0.00	
PALM	0.00	0.00	0.00	0.00	0.00	0.00	0.85	
STRUT	0.00	0.00	0.00	0.00	0.00	0.00	0.09	
THOUGHT	0.00	0.00	0.01	0.03	0.00	0.00	0.09	
TRAP	0.10	0.00	0.00	0.00	0.00	0.00	0.00	

In [25]:

```
pbcm
```

Out[25]:

	DRESS	FLEECE	FOOT	GOOSE	KIT	NURSE	PALM	ST
DRESS	0.658	0.013	0.004	0.001	0.237	0.011	0.000	(
FLEECE	0.006	0.962	0.000	0.000	0.031	0.000	0.000	(
FOOT	0.001	0.000	0.620	0.284	0.002	0.009	0.001	(
GOOSE	0.000	0.002	0.090	0.891	0.002	0.002	0.000	(
KIT	0.067	0.251	0.006	0.000	0.670	0.001	0.000	(
NURSE	0.040	0.003	0.030	0.007	0.041	0.866	0.000	(
PALM	0.002	0.000	0.006	0.001	0.000	0.000	0.550	(
STRUT	0.009	0.000	0.027	0.001	0.001	0.012	0.128	(
THOUGHT	0.000	0.000	0.130	0.059	0.000	0.000	0.059	(
TRAP	0.280	0.001	0.003	0.000	0.006	0.019	0.040	(

Next, let's analyze the confusion matrices in a couple different ways

- Take a simple difference

- Set up a dataframe to look at correlations
    - Convert matrices to stacksw
    - Join together into dataframe
- Calculate root mean square
- Get correlation using df.corr
- Set up a dataframe where all 0s are converted to NaN
    - Afterall, 0 correlates real good with 0!
- Get RMS
- Get correlation

In [26]:

```
cm - pbcm #differences
```

Out[26]:

vowelChoice	DRESS	FLEECE	FOOT	GOOSE	KIT	NURSE	PALM
DRESS	0.182	-0.013	-0.004	-0.001	-0.117	-0.001	0.00
FLEECE	-0.006	0.008	0.000	0.000	-0.001	0.000	0.00
FOOT	-0.001	0.000	0.200	-0.154	-0.002	-0.009	-0.00
GOOSE	0.000	-0.002	0.050	-0.071	-0.002	-0.002	0.00
KIT	0.003	-0.201	-0.006	0.000	0.210	-0.001	0.00
NURSE	0.020	-0.003	-0.030	-0.007	-0.021	0.024	0.00
PALM	-0.002	0.000	-0.006	-0.001	0.000	0.000	0.30
STRUT	-0.009	0.000	-0.027	-0.001	-0.001	-0.012	-0.03
THOUGHT	0.000	0.000	-0.120	-0.029	0.000	0.000	0.03
TRAP	-0.180	-0.001	-0.003	0.000	-0.006	-0.019	-0.04

In [27]:

```
# flatten and combine confusion matrices
pbcmfl = pd.Series(pbcm.stack(), name = "PB")
cmfl = pd.Series(cm.stack(), name = "GCM")
cms = pd.concat([pbcmfl, cmfl], axis = 1)

#Remove 0s (which correlate with each other)
cmsnan = cms.replace(0, np.nan)
```

In [28]:

```
print("0s included")
print("RMS =  ", (((cms.PB-cms.GCM) ** 2).mean()) ** .5)
print("r =    ", (cms['PB'].corr(cms['GCM'])))
print("")
print("0s removed")
print("RMS =  ",(((cmsnan.PB-cmsnan.GCM) ** 2).mean()) ** .5)
print("r =    ", cmsnan['PB'].corr(cmsnan['GCM']))
```

```
0s included
RMS =   0.0741105255682349
r =     0.964581333955327

0s removed
RMS =   0.12843085629429107
r =     0.9519894540508784
```

```
In [ ]:
```

Setting attention weights

In [1]:

```
%load_ext autoreload
from ExemPy import *
from ExemPy.utils import *
from ExemPy.viz import *
from ExemPy.GCM import *
%aimport ExemPy, ExemPy.utils, ExemPy.viz, ExemPy.GCM
%autoreload 1
import math
import random
import matplotlib.pyplot as plt
#%matplotlib inline
import numpy as np
import pandas as pd
from pandas import DataFrame
from scipy.optimize import minimize
import seaborn as sns
sns.set(style='ticks', context='paper')
colors=["#e3c934","#68c4bf","#c51000","#287271"]
sns.set_palette(colors)
```

In [2]:

```
# Read in data, set initial parameters
pb52 = pd.read_csv('pb52_data//pb52.csv')
pb52 = HzToBark(pb52, ["F0", "F1", "F2", "F3"])
dimsvals={'z0' : 1,
          'z1' : .761,
          'z2' : .681,
          'z3' : .407}
dimslist = list(dimsvals.keys())

catslist = ['type', 'vowel']

cval = 25

exemplars = pb52.copy()
```

In [9]:

```
# Define error function
def calcerror(x, test, exemplars, catslist, fitdims, cval, an
    '''
    Categorizes a data set and returns the proportion of stim
    rows that were categorized inaccurately. A lower value me
    lower amount of error. Designed to be used with parameter
    fitting functions to assign values to attention weighting
    for dimensions.

    Required paratemers:

    x = Array. Initial guesses for parameters

    test = DataFrame. Stimuli to be categorized

    exemplars = DataFrame. Exemplar cloud to use for categori

    catslist = List of strings. Each string should correspond
        category that should be assigned to the test

    fitdims = List of strings. Each string should correspond
        dimension for which parameters should be fit.

    Optional parameters:

    anchordim = String. Dimension for parameter which will no
        but will instead be hard-coded as 1. This helps const
        the set of possible solutions


    '''
    dimsvals = {fitdims[i]: x[i] for i in range(len(fitdims))
    if anchordim != None:
        dimsvals.update({anchordim:1})

    choices = multicat(test, cloud, catslist, dimsvals, cval)
    accuracy = checkaccuracy(choices, catslist)
    category = catslist[0]
    err = accuracy[category+"Acc"].value_counts(normalize=Tru
    return err
```

In [10]:

```
# Specify arguments for optimization
fitdims = dimslist[1:]      # Fit all dimensions except item
anchordim = dimslist[0]     # Set item 0 to 1

name = 'pb52-111723'        # name of output spreadsheet
nt = 3                      # number of times that random x i
t = 0.1                     # Tolerance value -- lower = more

# To demonstrate, fit based on 50 exemplars of each vowel
test = gettestset(exemplars, "vowel", 50)

cloud = exemplars
cats = ["vowel"]
```

In [7]:

```
# Optimize
# Initialize lists
resultslist=[['start','fit','error','evals']]
wlist=[]

print("----- Parameters -----")
if anchordim != None:
    print("Anchored (1):  ", anchordim)

print("Optimized:     ", fitdims)
print("")
print("Categorizing for: ", cats)
print("")
print("Trials: ", nt)
print("")

for i in range(0,nt):
    x=np.divide(random.sample(range(0,300),len(fitdims)),100)
    xguess = x
    result = minimize(calcerror,
                  xguess,
                  args=(test, cloud, cats, fitdims, cval, anc
                  method='Powell',
                  tol = t)
    # Create list to save as csv
    start = x
    fit = np.round(result.x,3)
    error = result.fun
    evals = result.nfev
    row = [start,fit,error,evals]
    resultslist.append(row)

    # Re-compose w dict to save with json
    wdict_keys = fitdims
    wdict_vals = list(fit)
    #if anchordim != None:
    wdict_keys.insert(0, anchordim)
    wdict_vals.insert(0, 'hi')
    wdict = {wdict_keys[i]: wdict_vals[i] for i in range(len(
    wlist.append(wdict)

    print ("-----", (i+1) ," -----")
    print("Initial guess:    ", start)
    print("Optimized:        ", fit)
    print(" ")
    print("Number evals: ", evals)
    print("Error:        ", error)
    print("")
```

```python
results=pd.DataFrame(resultslist)
results.columns = results.iloc[0]
results=results[1:]

settings = {"fitdims": fitdims, "anchordim": anchordim,
            "cats": cats, "trials":nt, "tol": t }

# Write results to csv
results.to_csv(name+".csv")
with open((name+"_info.txt"),"w") as file:
    file.write(str(settings))

#Clear lists
resultslist = []
```

```
----- Parameters -----
Anchored (1):   z0
Optimized:      ['z1', 'z2', 'z3']

Categorizing for:  ['vowel']

Trials:  3

----- 1  -----
Initial guess:     [2.53 0.1  0.28]
Optimized:         [ 1.912  0.482 -0.192]

Number evals:  33
Error:         0.1

----- 2  -----
Initial guess:     [1.34 2.29 1.85 0.05]
Optimized:         [3.092 0.987 0.633 0.585]

Number evals:  52
Error:         0.098

----- 3  -----
Initial guess:     [1.15 0.56 2.44 1.38 2.32]
Optimized:         [2.15  1.56  3.44  1.526 2.367]

Number evals:  28
Error:         0.118
```

Correlations

In [12]:

```
%load_ext autoreload
from ExemPy import *
from ExemPy.utils import *
from ExemPy.viz import *
from ExemPy.GCM import *
%aimport ExemPy, ExemPy.utils, ExemPy.viz, ExemPy.GCM
%autoreload 1
import math
import random
import matplotlib.pyplot as plt
#%matplotlib inline
import numpy as np
import pandas as pd
from pandas import DataFrame
from scipy.optimize import minimize
import seaborn as sns
sns.set(style='ticks', context='paper')
colors=["#e3c934","#68c4bf","#c51000","#287271"]
sns.set_palette(colors)
```

```
The autoreload extension is already loaded. To reload it, use:
  %reload_ext autoreload
```

In [13]:

```
# Read in data and confusion matrix
pb52 = pd.read_csv('pb52_data//pb52.csv')
pbcm = pd.read_csv('pb52_data//pbcm.csv').drop([0]).set_index
    'vowelChoice').rename_axis(None)
pb52 = HzToBark(pb52, ["F0", "F1", "F2", "F3"])
pb52.sample(5)
```

Out[13]:

	type	gender	speaker	vowel	repetition	F0	F1	F2	
362	m	m	19	KIT	1	132	370	1750	27
1458	c	f	73	NURSE	1	300	540	1770	20
289	m	m	15	STRUT	2	110	660	960	24
454	m	m	23	FOOT	1	125	390	900	2
417	m	m	21	GOOSE	2	145	290	1000	23

In [14]:

```
# Set parameters
catslist = ["type", "vowel"]
```

```
cval = 25
exemplars = pb52
test = pb52
```

In [15]:
```
dims1 = {'z0' : 1,
         'z1' : 3.585,
         'z2' : 2.246,
         'z3' : 2.736}

dims2 = {'z0' : 1,
         'z1' : 2.72,
         'z2' : 1.322,
         'z3' : 0.882}

dims3 = {'z0' : 1,
         'z1' : 1.589,
         'z2' : 0.586,
         'z3' : 0.55}

dims4 = {'z0' : 1,
         'z1' : 2.534,
         'z2' : 1.891,
         'z3' : 1.784}

dims5 = {'z0' : 1,
         'z1' : 1.685,
         'z2' : 1.59,
         'z3' : 1.162}
```

In [16]:
```
weightslist = [dims1, dims2, dims3, dims4, dims5]
```

Next, let's categorize!

This may take a minute!

In [17]:
```
# Flatten the confusion matrix for comparison
pbflat = pd.Series(pbcm.stack(), name = "PB")
```

In [18]:
```
i = 1
rowlist=[]

print("trial", "rms PB", "r PB", "rms HG", "r HG")
for w in weightslist:
    choices = multicat(
        testset = test,
        cloud = exemplars,
        cats = catslist,
        dimsdict = w,
        c = cval,
        N = 1,
        exclude_self = True)

    cm = confusion(choices, catslist)['vowel']
    flat = pd.Series(cm.stack(), name = "GCM")
    matrices = pd.concat([pbflat, flat], axis=1)
    matrices.replace(0, np.nan)

    rms = (((matrices.PB-matrices.GCM) ** 2).mean()) ** .5
    r = matrices['PB'].corr(matrices['GCM'])

    matrices2 = pd.concat([hgflat, flat], axis=1)
    matrices2.replace(0, np.nan)

    rms2 = (((matrices2.HG-matrices2.GCM) ** 2).mean()) ** .5
    r2 = matrices2['HG'].corr(matrices2['GCM'])

    print(i, np.round(rms,3), np.round(r,3), np.round(rms2,3)

    i += 1
```

```
trial rms PB r PB rms HG r HG
1 0.073 0.964 0.073 0.964
2 0.075 0.964 0.075 0.964
3 0.074 0.966 0.074 0.966
4 0.073 0.965 0.073 0.965
5 0.075 0.966 0.075 0.966
```

Simulating forced choice tasks in ExemPy

ExemPy is built to simulate the types of tasks that are often used in speech perception experiments. This allows to more directly compare simulated results with behavior.

In ExemPy-Basics, we simulated an identifcation task, in which the "perceiver" provides a label from multiple options. This allowed us to produce a confusion matrix.

Simulating a two-alternative forced choice (2AFC) task allows us to produce the type of plot often used to demonstrate categorical perception.

In this section we'll

1. Create a continuum interpolated from one vowel to another
2. Categorize that continuum
3. Introduce considerations for simulating forced choice
4. Visualize forced choice results
5. Repeat the process with additional continua
6. Analyze the graph

In [16]:

```
%load_ext autoreload
from ExemPy import *
from ExemPy.utils import *
from ExemPy.viz import *
from ExemPy.GCM import *
%aimport ExemPy, ExemPy.utils, ExemPy.viz, ExemPy.GCM
%autoreload 1
import math
import random
import matplotlib.pyplot as plt
#%matplotlib inline
import numpy as np
import pandas as pd
from pandas import DataFrame
from scipy.optimize import minimize
```

```
import seaborn as sns
sns.set(style='ticks', context='paper')
colors=["#e3c934","#68c4bf","#c51000","#287271"]
sns.set_palette(colors)
```

```
The autoreload extension is already loaded. To reload it, use:
  %reload_ext autoreload
```

Set up data

- Read in Peterson and Barney 1952
- Convert Hz to Bark
- Preview pb52
- Set parameters for preliminary categorization

In [17]:

```
pb52 = pd.read_csv('pb52_data//pb52.csv')
pb52 = HzToBark(pb52,["F0", "F1", "F2", "F3"])
excloud = pb52.copy()
```

In [18]:

```
dimsvals = {'z0' : 1,
            'z1' : .761,
            'z2' : .681,
            'z3' : .407}
dimslist = list(dimsvals.keys())
catslist = ['type', 'vowel']          # man, woman, or child
cval = 25
```

Create type m continuum

Get mean dimensions for each type x vowel combination

In [19]:

```
datasumm = datasummary(pb52, catslist, dimslist)
print("Type 'm' formant averages (Bark) by vowel:")
print(datasumm[datasumm["type"] == "m"])
```

```
Type 'm' formant averages (Bark) by vowel:
   type    vowel        z0        z1         z2         z3
10    m    DRESS  1.116743  5.132435  12.491891  14.438514
11    m   FLEECE  1.207534  2.674882  13.914472  15.535502
12    m     FOOT  1.208595  4.347787   8.636553  13.766756
13    m    GOOSE  1.247804  3.093098   7.695949  13.727507
14    m      KIT  1.204788  3.931608  12.975164  14.668190
15    m    NURSE  1.168517  4.807455  10.439210  11.939155
16    m     PALM  1.061577  6.653223   9.038935  14.315383
17    m    STRUT  1.118489  5.989758   9.588147  14.146945
18    m  THOUGHT  1.100155  5.484137   7.457763  14.207406
19    m     TRAP  1.073975  6.240755  12.013320  14.270487
```

Create a FOOT-STRUT continuum using m values

In [20]:

```
start = {"type" : "m", "vowel" : "FOOT"}      # Step 1
end = {"type" : "m", "vowel" : "STRUT"}      # Step n
steps = 7      # Number of steps in continuum

mcont = continuum(datasumm,      # dataframe for starting val
                 start,
                 end,
                 dimslist,
                 steps,
                 stimdetails = True      # Print details of t
                )
print(mcont)
```

```
Start:  index          12
type            m
vowel        FOOT
z0       1.208595
z1       4.347787
z2       8.636553
z3      13.766756
Name: 0, dtype: object
End:  index          17
type            m
vowel       STRUT
z0       1.118489
z1       5.989758
z2       9.588147
z3      14.146945
Name: 0, dtype: object
   step        z0        z1        z2         z3
0     1  1.208595  4.347787  8.636553  13.766756
1     2  1.193577  4.621449  8.795152  13.830121
2     3  1.178560  4.895111  8.953751  13.893486
3     4  1.163542  5.168773  9.112350  13.956851
4     5  1.148524  5.442435  9.270949  14.020215
5     6  1.133507  5.716097  9.429548  14.083580
6     7  1.118489  5.989758  9.588147  14.146945
```

What does forced choice mean for perception?

During a forced choice task, an experiment participant is, well, forced to choose between two options. Often, these choices are two clear endpoints of an interpolated continuum like the one we just created.

But what does it mean for a perceiver to know that they'll have to pick between two choices?

At this time, I identify three, non mutually exclusive vectors for modelling the "side effects" of a forced choice task. These occur at two different levels: the level of perception and the level of deciding how to respond.

My thinking about this dinstinction is heavily influenced by Zheng and Samuel's (2017) problematization of *perception* vs *interpretation*. I need to re-read the paper to decide/interpret whether I'm using the word decision is a different way than they use interpretation. To be honest, I definitely thought they had said "decide" until I looked up the citation. It almost feels like a "below vs above the level of consciousness" situation.

1. At the level of perception (implemented during the `activation()` function)

- Base activation *N*: The expectation within the experiment is that the percept will be one of two options. Activation may be raised for the alternatives and quashed for would-be competitors in a straightforward example of priming.
- Attention weights *w*:

2. At the level of decision (implemented during the `choose()` function)

- Rather than choosing the label with the highest probability overall, the 'perceiver' chooses the alternative with the higher probability of the two

The argument `fc = {<category type> : [<label>, <label>, ...]}` implements forced choice at the level of decision. Probabilities for the category type given are calculated as usual, but the choices are restricted to the options given.

It's tempting to see the `activation()` function as 'perception' and the `choose()` function as interpretation. At every step, I think we should be explicit about the following: Something making the implementation possible, or being possible in the implementation, does not imply the analog in language users' perception or interpretation.

Categorize continuum with respect to pb52

```
In [21]: terms = {"vowel" : ["FOOT", "STRUT"]}
```

In [22]:

```
choices_mcont = multicat(
    mcont,
    excloud,
    catslist,
    dimsvals,
    cval,
    exclude_self = True,
    N = 1,
    fc = terms)
print(choices_mcont)
```

```
   step        z0        z1        z2         z3 typeChoice  t
ypeProb  \
0     1  1.208595  4.347787  8.636553  13.766756          m  0
.998427
1     2  1.193577  4.621449  8.795152  13.830121          m  0
.998456
2     3  1.178560  4.895111  8.953751  13.893486          m  0
.998329
3     4  1.163542  5.168773  9.112350  13.956851          m  0
.995452
4     5  1.148524  5.442435  9.270949  14.020215          m  0
.998454
5     6  1.133507  5.716097  9.429548  14.083580          m  0
.999070
6     7  1.118489  5.989758  9.588147  14.146945          m  0
.999296

  vowelChoice  vowelProb
0        FOOT   0.975880
1        FOOT   0.994435
2        FOOT   0.972424
3       STRUT   0.555092
4       STRUT   0.978905
5       STRUT   0.976005
6       STRUT   0.943317
```

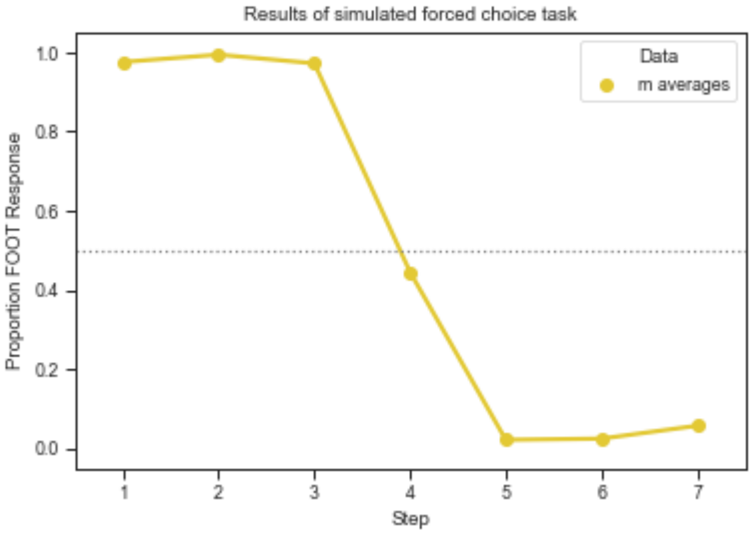
Categorical perception plot

The plot below shows the results of our simulated 2AFC identification task. On the X-axis is the continuum from FOOT (step 1) to STRUT (step 7). On the Y-axis is the probability of categorization as FOOT (rather than STRUT), which is taken as analagous to the proportion of FOOT responses in an experiment.

A dotted line is printed at 0.5, or roughly chance. Above this line, "FOOT" is a more likely response than "STRUT". We can consider the point along the continuum at which the curve crosses this line as the boundary between "FOOT" and "STRUT".

In [23]:

```
p = cpplot(datalist = choices_mcont,
           cat = "vowel",
           xax = "step",
           datanames = ["m averages"]
          )
p.set(title = "Results of simulated forced choice task")
plt.show()
```



Repeat this process with average woman values

Repeating the process with the average woman values demonstrates the need for care and transparency in how the "forced choice" aspect of the text is simulated.

For the man continuum, the probability was always above 0.5. For the w values, this won't be the case.

In [24]:

```
start = {"type" : "w", "vowel" : "FOOT"}      # Step 1
end = {"type" : "w", "vowel" : "STRUT"}      # Step n
steps = 7      # Number of steps in continuum

wcont = continuum(datasumm, start, end, dimslist, steps, stim
choices_wcont = multicat(
    wcont,
    excloud,
    catslist,
    dimsvals,
    cval,
    exclude_self = True,
    N = 1,
    fc = terms)
print(choices_wcont)
```

```
   step       z0        z1         z2         z3 typeChoice  t
ypeProb  \
0     1  2.32341  4.636707   9.405672  14.947067          w  0
.958173
1     2  2.30292  5.019998   9.615083  14.981178          w  0
.977810
2     3  2.28243  5.403288   9.824495  15.015290          w  0
.980696
3     4  2.26194  5.786579  10.033906  15.049401          w  0
.960333
4     5  2.24145  6.169870  10.243318  15.083513          w  0
.961737
5     6  2.22096  6.553161  10.452729  15.117624          w  0
.990411
6     7  2.20047  6.936452  10.662141  15.151736          w  0
.983766

  vowelChoice  vowelProb
0        FOOT   0.974100
1        FOOT   0.992542
2        FOOT   0.607590
3        FOOT   0.280120
4       STRUT   0.767191
5       STRUT   0.994830
6       STRUT   0.994342
```

At step 4, there is only a 0.28 probability that the vowel will be classified as FOOT. Although we expect to see uncertainty at the most ambiguous point along the continuum, this is an unusually low value for any winning category label.

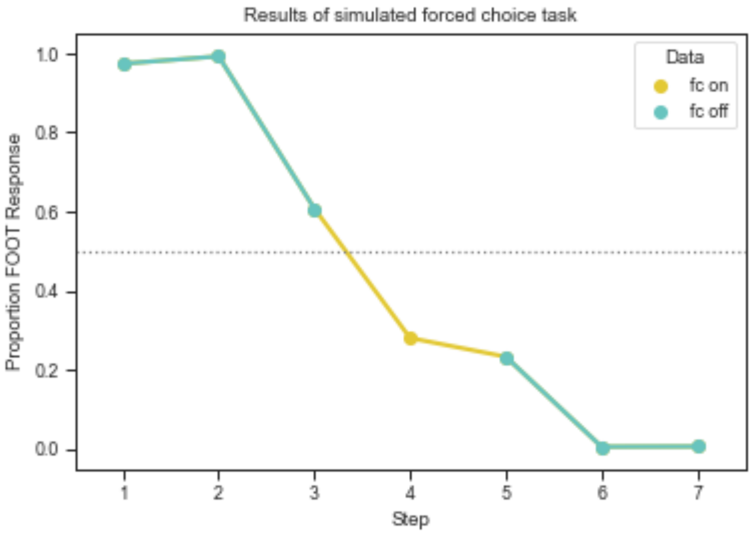
In fact, PALM, not FOOT, was the label with the highest probability. To see this, let's turn off the forced choice parameter with `fc = None` (or by simply not providing the argument).

```
In [25]: choices_wcont_2 = multicat(
             wcont,
             excloud,
             catslist,
             dimsvals,
             cval,
             exclude_self = True,
             N = 1,
             fc = None)      # Turn off the forced choice terms
         print(choices_wcont_2)
```

```
   step       z0        z1         z2         z3 typeChoice  t
ypeProb  \
0     1  2.32341  4.636707   9.405672  14.947067          w  0
.958173
1     2  2.30292  5.019998   9.615083  14.981178          w  0
.977810
2     3  2.28243  5.403288   9.824495  15.015290          w  0
.980696
3     4  2.26194  5.786579  10.033906  15.049401          w  0
.960333
4     5  2.24145  6.169870  10.243318  15.083513          w  0
.961737
5     6  2.22096  6.553161  10.452729  15.117624          w  0
.990411
6     7  2.20047  6.936452  10.662141  15.151736          w  0
.983766

  vowelChoice  vowelProb
0        FOOT   0.974100
1        FOOT   0.992542
2        FOOT   0.607590
3        PALM   0.624499
4       STRUT   0.767191
5       STRUT   0.994830
6       STRUT   0.994342
```

```
In [26]: p = cpplot([choices_wcont, choices_wcont_2],
                    "vowel",
                    xax = "step",
                    datanames = ["fc on", "fc off"])
         p.set(title = "Results of simulated forced choice task")
         plt.show()
```

```
----- Hey! -----
Stimulus 4
in dataset fc off
was categorized as PALM
with probability 0.62
```



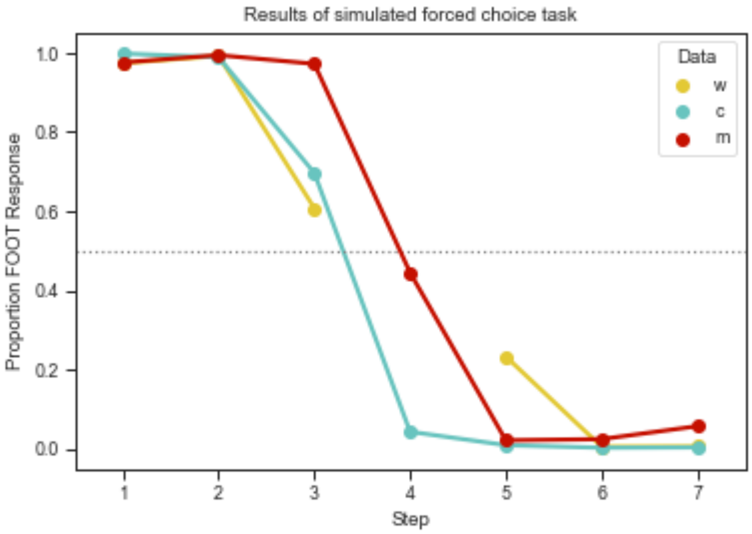
Repeat the process for all 3 types

Next

In [27]:
```
continuum_is = {"vowel" : ["FOOT", "STRUT"]}
grouping = {"type" : ["w", "c", "m"]}
steps = 7
dimsvals = {'z0' : 1,
            'z1' : .761,
            'z2' : .681,
            'z3' : .407}
dimslist = list(dimsvals.keys())
catslist = ['vowel']
cval = 25
excloud = pb52.copy()
```

In [28]:
```
# unpack variables
contcat = list(continuum_is.keys())[0]
start = continuum_is[contcat][0]
end = continuum_is[contcat][1]
groupingcat = list(grouping.keys())[0]
labellist = grouping[groupingcat]

datsumm = datasummary(excloud, ['type','vowel'], dimslist)
datalist = []

for l in labellist:
    # get start and end rows
    strt = {contcat : start, groupingcat : l}
    nd = {contcat : end, groupingcat : l}
    # make continuum
    cnt = continuum(datsumm, strt, nd, dimslist, steps = step
    # make choices
    chs = multicat(cnt, excloud, catslist, dimsvals, cval,
                  exclude_self = True, N = 1, fc = None)
    datalist.append(chs)

p = cpplot(datalist = datalist,
           cat = "vowel",
           xax = "step",
           datanames = labellist)
p.set(title = "Results of simulated forced choice task")
plt.show()
```

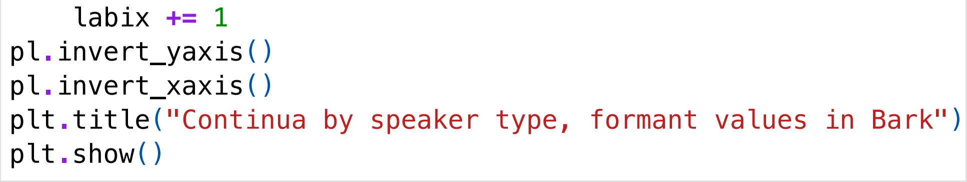
```
----- Hey! -----
Stimulus 4
in dataset w
was categorized as PALM
with probability 0.62
```



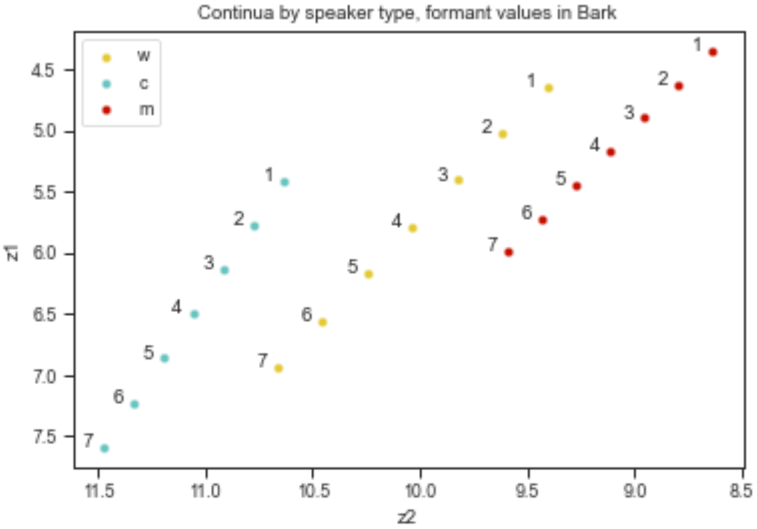
And let's visualize the continua for good measure

In [29]:

```
labix = 0
for d in datalist:
    # get the name of the dataset
    lab = labellist[labix]
    pl = sns.scatterplot(x = 'z2', y = 'z1', data = d, label
    # Label points
    for line, row in d.iterrows():
        pl.text(d['z2'][line]+0.1, d['z1'][line], d['step'][l
```



```
    labix += 1
pl.invert_yaxis()
pl.invert_xaxis()
plt.title("Continua by speaker type, formant values in Bark")
plt.show()
```



Priming and Resonance

In [15]:

```
%load_ext autoreload
import ExemPy
from ExemPy import *
from ExemPy.utils import *
from ExemPy.viz import *
from ExemPy.GCM import *
%aimport ExemPy, ExemPy.utils, ExemPy.viz, ExemPy.GCM
%autoreload 1
import math
import random
import matplotlib.pyplot as plt
#%matplotlib inline
import numpy as np
import pandas as pd
from pandas import DataFrame
from scipy.optimize import minimize
import seaborn as sns
sns.set(style='ticks', context='notebook')
colors=["#e3c934","#68c4bf","#c51000","#287271"]
sns.set_palette(colors)
```

```
The autoreload extension is already loaded. To reload it, use:
  %reload_ext autoreload
```

In [16]:

```
# Read in data
pb52 = pd.read_csv('pb52_data//pb52.csv')
pb52 = HzToBark(pb52,["F0", "F1", "F2", "F3"])
```

In [17]:

```
# Set parameters
w1 = {'z0' : 0.10, 'z1' : 0.37, 'z2' : 0.23, 'z3' : 0.29}
w2 = {'z0' : 0.17, 'z1' : 0.46, 'z2' : 0.22, 'z3' : 0.15}
w3 = {'z0' : 0.27, 'z1' : 0.16, 'z2' : 0.15, 'z3' : 0.25}
w4 = {'z0' : 0.14, 'z1' : 0.35, 'z2' : 0.26, 'z3' : 0.21}
w5 = {'z0' : 1, 'z1' : .761, 'z2' : .681, 'z3' : .407}

wlist = [w1, w2, w3, w4, w5]


dimsvals = w1
dimslist = list(dimsvals.keys())
catslist = ['type', 'vowel']
cval = 25
```

```
exemplars = pb52
test = pb52
```

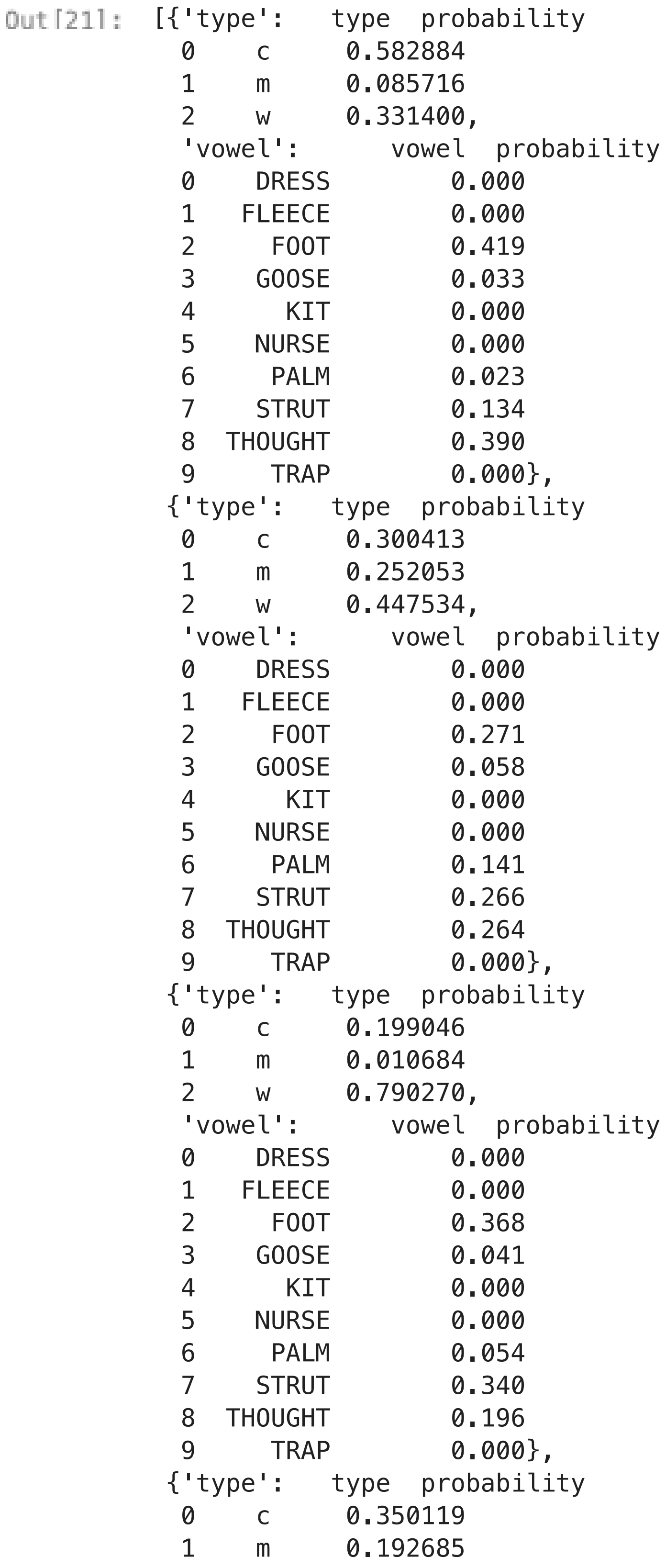
```
In [18]: thestim = pb52.loc[(pb52['speaker']==47) & (pb52['vowel']=='F
```

```
In [19]: print(thestim)
```

```
    type gender  speaker vowel  repetition   F0   F1    F2  
F3        z0  \
935    w      f       47  FOOT           2  205  570  1200  29
70  2.008591   

           z1        z2         z3  
935  5.510198  9.651013  15.621258  
```

```
In [20]: # Get probabilities in normal categorization task
prlist = []
choicelist = []
for w in wlist:
    exemplars = exclude(exemplars, thestim,
                        exclude_self = True,
                        alsoexclude='speaker')
    exemplars = reset_N(exemplars, N = 1)
    bigdf = activation(thestim, exemplars,
                       dimsdict = w, c = cval)
    pr = probs(bigdf, catslist)
    pr['vowel']=np.round(pr['vowel'],3)
    prlist.append(pr)
    choices = choose(pr, thestim, catslist)
    choicelist.append(choices)
```

```
In [21]: prlist
```

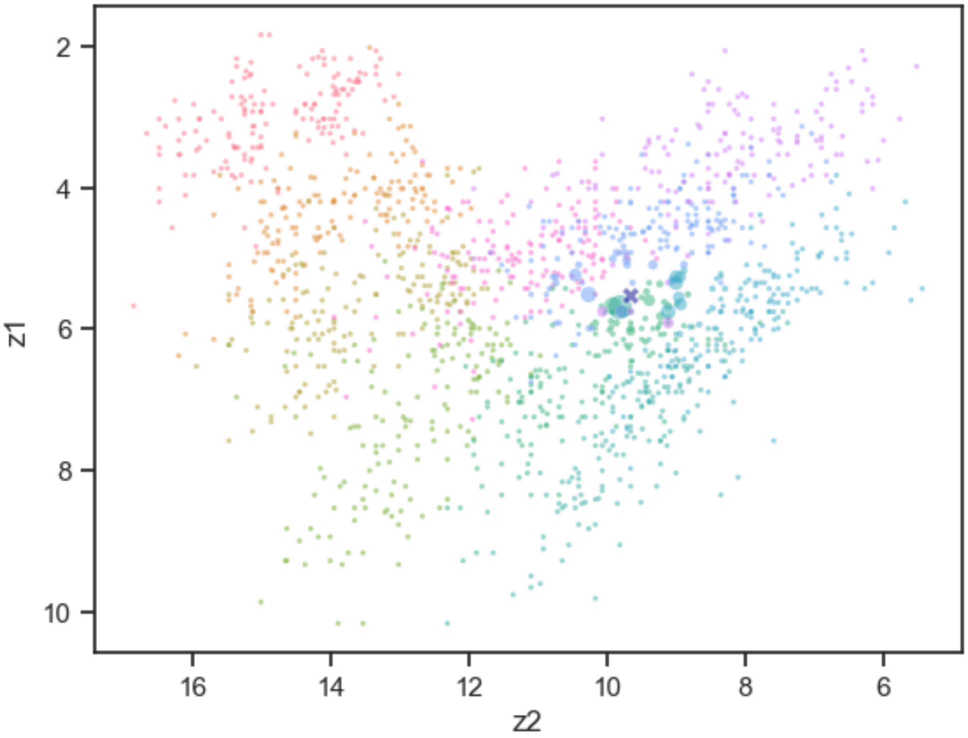


```
Out[21]: [{'type':   type  probability
           0    c     0.582884
           1    m     0.085716
           2    w     0.331400,
           'vowel':      vowel  probability
           0    DRESS        0.000
           1   FLEECE        0.000
           2     FOOT        0.419
           3    GOOSE        0.033
           4      KIT        0.000
           5    NURSE        0.000
           6     PALM        0.023
           7    STRUT        0.134
           8  THOUGHT        0.390
           9     TRAP        0.000},
          {'type':   type  probability
           0    c     0.300413
           1    m     0.252053
           2    w     0.447534,
           'vowel':      vowel  probability
           0    DRESS        0.000
           1   FLEECE        0.000
           2     FOOT        0.271
           3    GOOSE        0.058
           4      KIT        0.000
           5    NURSE        0.000
           6     PALM        0.141
           7    STRUT        0.266
           8  THOUGHT        0.264
           9     TRAP        0.000},
          {'type':   type  probability
           0    c     0.199046
           1    m     0.010684
           2    w     0.790270,
           'vowel':      vowel  probability
           0    DRESS        0.000
           1   FLEECE        0.000
           2     FOOT        0.368
           3    GOOSE        0.041
           4      KIT        0.000
           5    NURSE        0.000
           6     PALM        0.054
           7    STRUT        0.340
           8  THOUGHT        0.196
           9     TRAP        0.000},
          {'type':   type  probability
           0    c     0.350119
           1    m     0.192685
```

```
  2    w     0.457196,
  'vowel':      vowel  probability
  0    DRESS        0.000
  1   FLEECE        0.000
  2     FOOT        0.369
  3    GOOSE        0.053
  4      KIT        0.000
  5    NURSE        0.000
  6     PALM        0.082
  7    STRUT        0.270
  8  THOUGHT        0.226
  9     TRAP        0.000},
 {'type':   type  probability
  0    c     0.136430
  1    m     0.033758
  2    w     0.829812,
  'vowel':      vowel  probability
  0    DRESS        0.000
  1   FLEECE        0.000
  2     FOOT        0.440
  3    GOOSE        0.044
  4      KIT        0.000
  5    NURSE        0.000
  6     PALM        0.073
  7    STRUT        0.370
  8  THOUGHT        0.073
  9     TRAP        0.000}]
```

In [22]:

```
# Get activation plot for normal categoriztaion
exemplars = exclude(exemplars, thestim,
                    exclude_self = True,
                    alsoexclude='speaker')
exemplars = reset_N(exemplars, N = 1)
bigdf = activation(thestim, exemplars,
                   dimsdict = w2, c = cval)
a = getactiv(activxn = bigdf,
             x='z2', y = 'z1',
             cat='vowel')
pl = activplot(a=a, x='z2', y='z1',
               cat='vowel', test=thestim,
               invert = True)
```



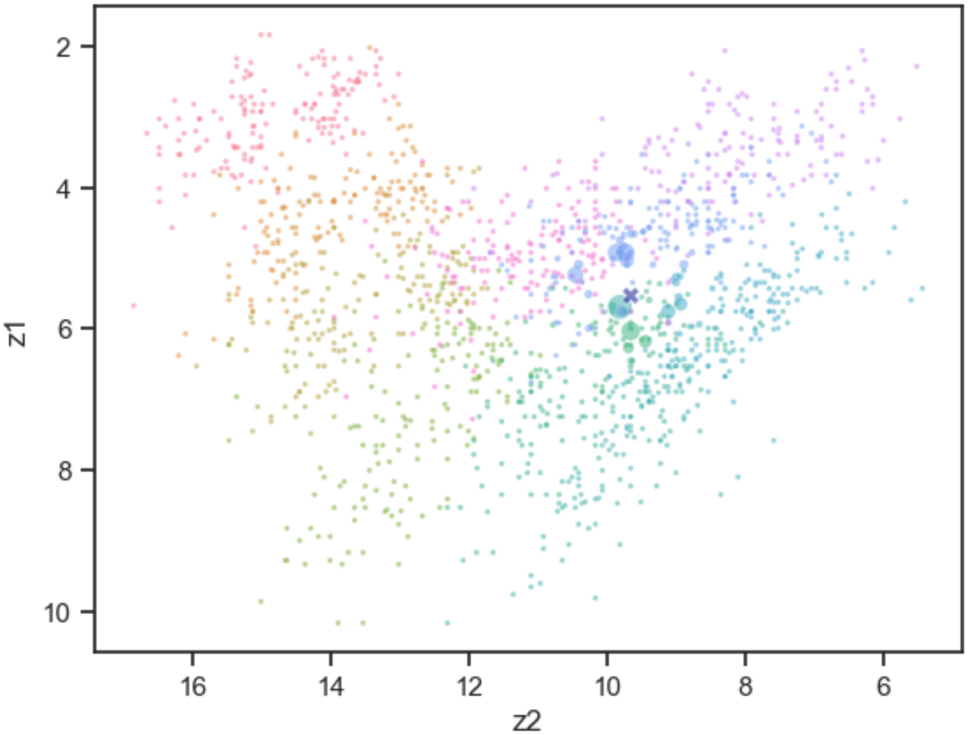
```
In [23]: # Set up dicts to refer to later
         typebias = {"m":1,
                     "c":1,
                     "w":10}
         vowelbias = {"DRESS" : 1,
                      "FLEECE" : 1,
                      "FOOT" : 100,
                      "GOOSE" : 1,
                      "KIT" : 1,
                      "NURSE" : 1,
                      "PALM" : 1,
                      "STRUT" : 1,
                      "THOUGHT" : 1,
                      "TRAP" : 1}
```

```
In [24]: cat = "type"
         catbias = typebias
```

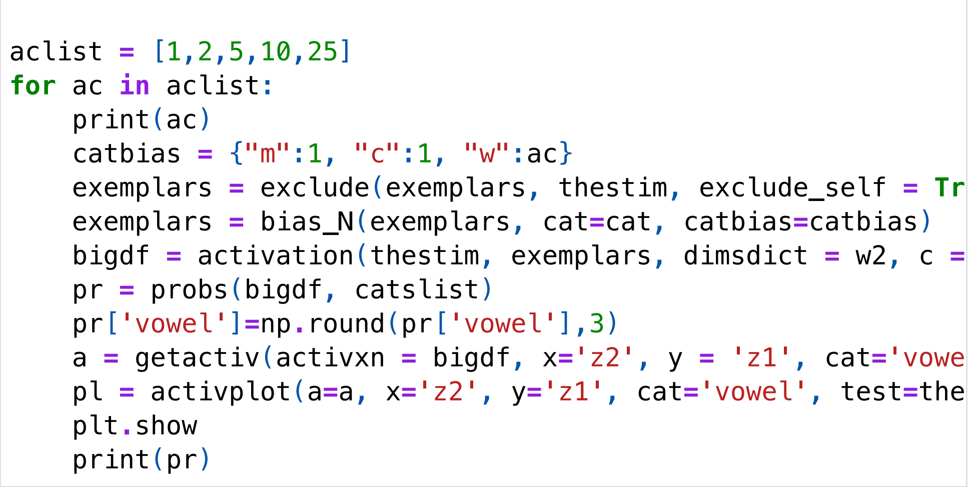
```
In [25]: # Get probabilities where N for w = 10
         exemplars = exclude(exemplars, thestim,
                             exclude_self = True,
                             alsoexclude='speaker')
         exemplars = bias_N(exemplars,
                            cat=cat, catbias=catbias)
```

```
bigdf = activation(thestim, exemplars,
                   dimsdict = w4, c = cval)
pr = probs(bigdf, catslist)
pr['vowel']=np.round(pr['vowel'],3)
a = getactiv(activxn = bigdf, x='z2', y = 'z1', cat='vowel')
pl = activplot(a=a, x='z2', y='z1', cat='vowel',
               test=thestim, invert = True)
print(pr)
```

```
{'type':   type  probability
0    c     0.068453
1    m     0.037672
2    w     0.893875, 'vowel':      vowel  probability
0    DRESS        0.000
1   FLEECE        0.000
2     FOOT        0.474
3    GOOSE        0.012
4      KIT        0.000
5    NURSE        0.000
6     PALM        0.153
7    STRUT        0.201
8  THOUGHT        0.160
9     TRAP        0.000}
```

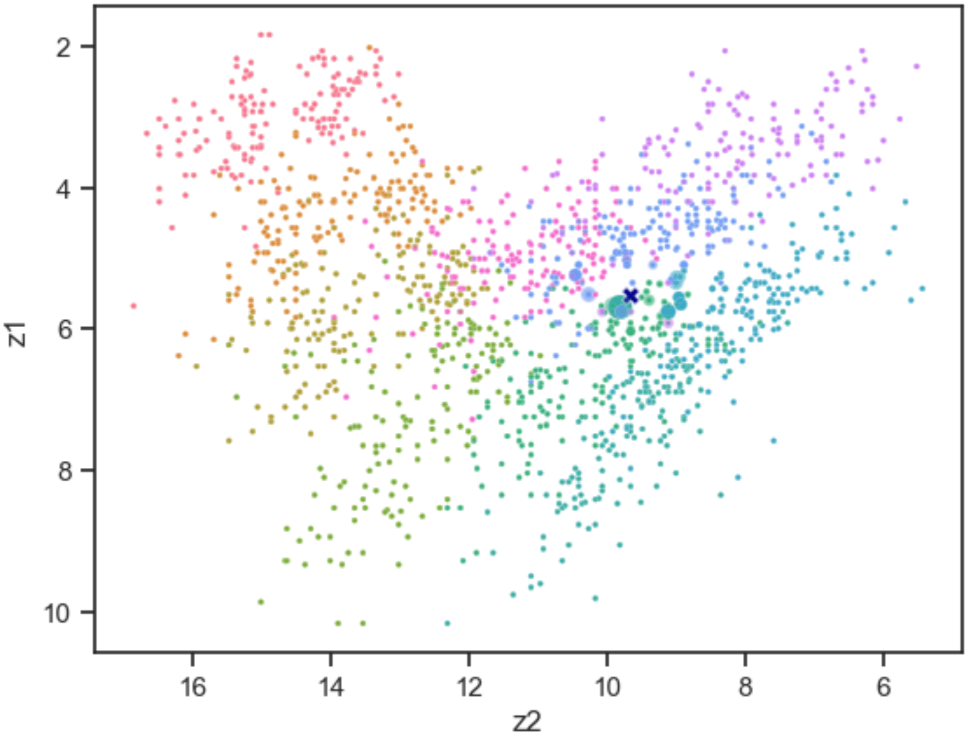


```
In [30]: # Try out different numbers of bias towards women
         cat = "type"
```



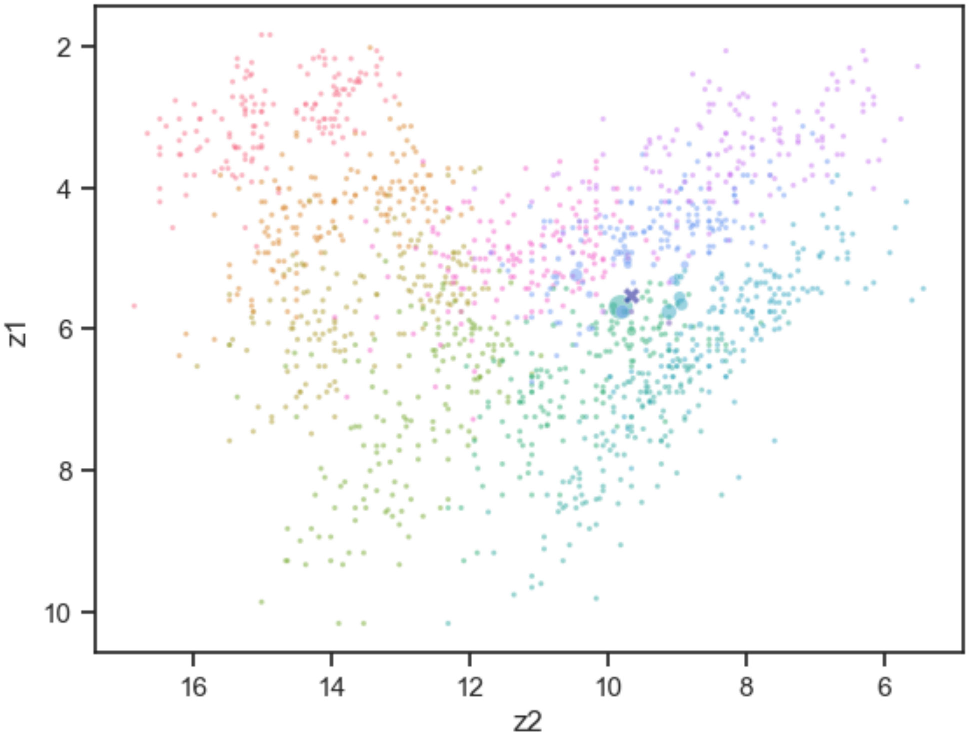
```
aclist = [1,2,5,10,25]
for ac in aclist:
    print(ac)
    catbias = {"m":1, "c":1, "w":ac}
    exemplars = exclude(exemplars, thestim, exclude_self = Tr
    exemplars = bias_N(exemplars, cat=cat, catbias=catbias)
    bigdf = activation(thestim, exemplars, dimsdict = w2, c =
    pr = probs(bigdf, catslist)
    pr['vowel']=np.round(pr['vowel'],3)
    a = getactiv(activxn = bigdf, x='z2', y = 'z1', cat='vowe
    pl = activplot(a=a, x='z2', y='z1', cat='vowel', test=the
    plt.show
    print(pr)
```

```
1
{'type':   type  probability
0    c     0.300413
1    m     0.252053
2    w     0.447534, 'vowel':      vowel  probability
0    DRESS        0.000
1   FLEECE        0.000
2     FOOT        0.271
3    GOOSE        0.058
4      KIT        0.000
5    NURSE        0.000
6     PALM        0.141
7    STRUT        0.266
8  THOUGHT        0.264
9     TRAP        0.000}
2
{'type':   type  probability
0    c     0.207534
1    m     0.174126
2    w     0.618340, 'vowel':      vowel  probability
0    DRESS        0.000
1   FLEECE        0.000
2     FOOT        0.293
3    GOOSE        0.040
4      KIT        0.000
5    NURSE        0.000
6     PALM        0.193
7    STRUT        0.199
8  THOUGHT        0.274
9     TRAP        0.000}
5
{'type':   type  probability
0    c     0.107670
1    m     0.090337
2    w     0.801993, 'vowel':      vowel  probability
0    DRESS        0.000
1   FLEECE        0.000
2     FOOT        0.317
3    GOOSE        0.021
4      KIT        0.000
5    NURSE        0.000
6     PALM        0.249
7    STRUT        0.127
8  THOUGHT        0.285
9     TRAP        0.000}
10
{'type':   type  probability
0    c     0.059750
1    m     0.050132
2    w     0.890118, 'vowel':      vowel  probability
0    DRESS        0.000
1   FLEECE        0.000
2     FOOT        0.328
3    GOOSE        0.012
4      KIT        0.000
5    NURSE        0.000
6     PALM        0.276
7    STRUT        0.093
8  THOUGHT        0.290
9     TRAP        0.000}
25
{'type':   type  probability
0    c     0.025587
1    m     0.021468
2    w     0.952945, 'vowel':      vowel  probability
0    DRESS        0.000
1   FLEECE        0.000
2     FOOT        0.336
3    GOOSE        0.005
4      KIT        0.000
5    NURSE        0.000
6     PALM        0.296
7    STRUT        0.069
8  THOUGHT        0.294
9     TRAP        0.000}
```

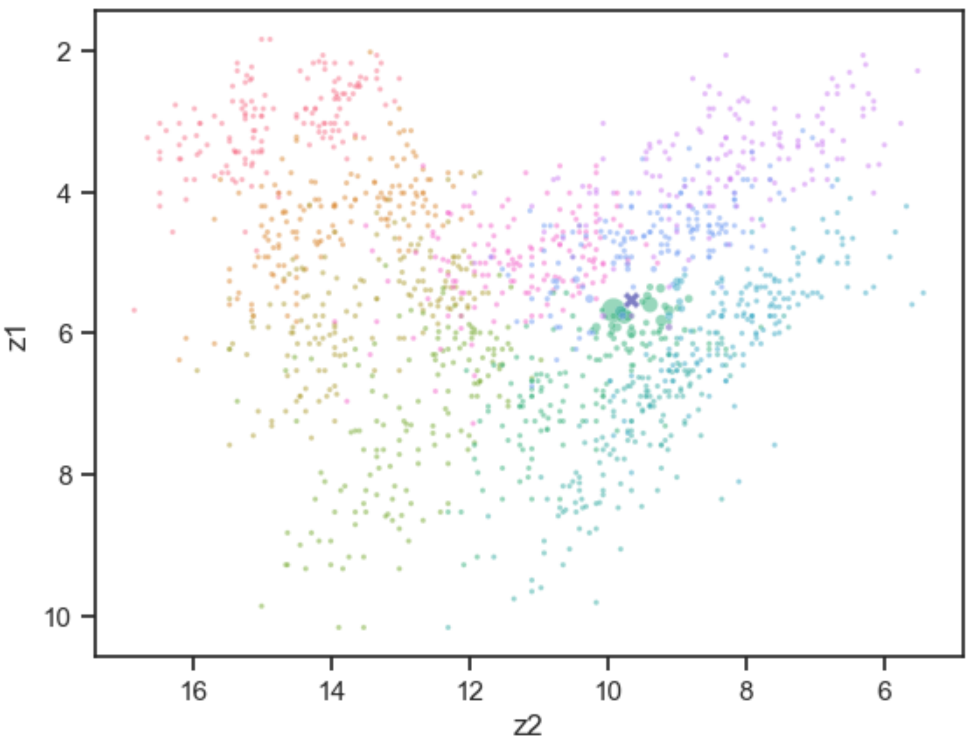


```
In [32]: # Bias towards women
         cat = "type"
         catbias = {"m":1, "c":1, "w":10}
         exemplars = exclude(exemplars, thestim,
                             exclude_self = True,
                             alsoexclude='speaker')
         exemplars = bias_N(exemplars, cat=cat,
                            catbias=catbias)
         bigdf = activation(thestim, exemplars,
                            dimsdict = w2, c = cval)
         pr = probs(bigdf, catslist)
         pr['vowel']=np.round(pr['vowel'],3)
         a = getactiv(activxn = bigdf, x='z2', y = 'z1',
                      cat='vowel')
         pl = activplot(a=a, x='z2', y='z1', cat='vowel',
                        test=thestim, invert = True)
         plt.show
         #print(pr)
```

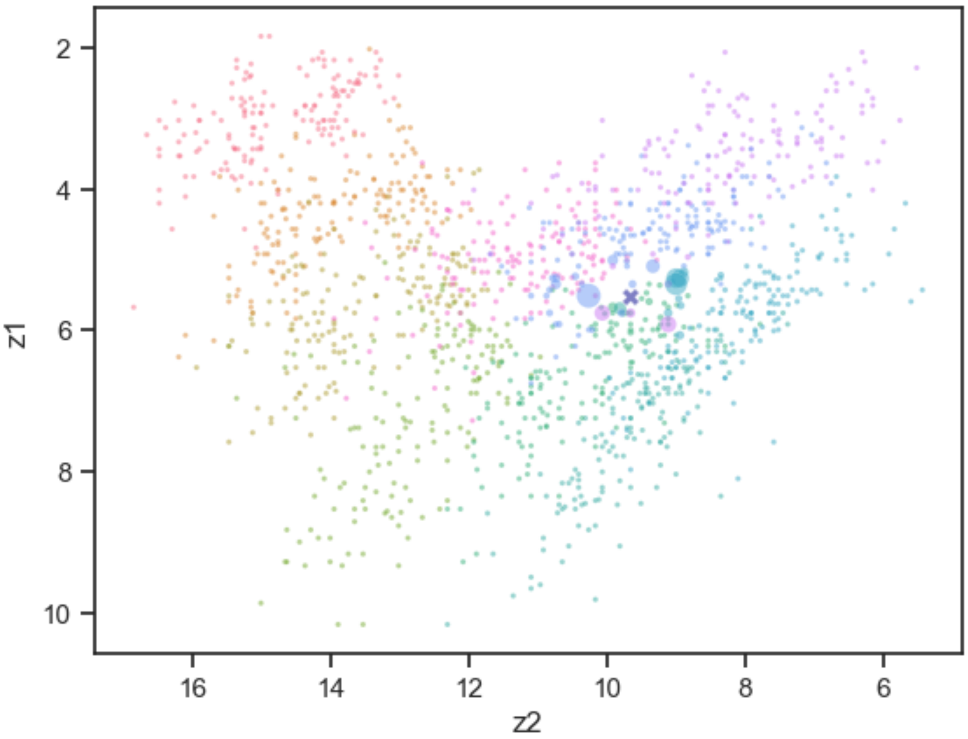
```
Out[32]: <function matplotlib.pyplot.show(close=None, block=None)>
```



```
In [33]: # Bias towards men
         cat = "type"
         catbias = {"m":10, "c":1, "w":1}
         exemplars = exclude(exemplars, thestim,
                             exclude_self = True,
                             alsoexclude='speaker')
         exemplars = bias_N(exemplars, cat=cat,
                            catbias=catbias)
         bigdf = activation(thestim, exemplars,
                            dimsdict = w2, c = cval)
         pr = probs(bigdf, catslist)
         pr['vowel']=np.round(pr['vowel'],3)
         a = getactiv(activxn = bigdf, x='z2', y = 'z1',
                      cat='vowel')
         pl = activplot(a=a, x='z2', y='z1', cat='vowel',
                        test=thestim, invert = True)
         plt.show
         #print(pr)

Out[33]: <function matplotlib.pyplot.show(close=None, block=None)>
```



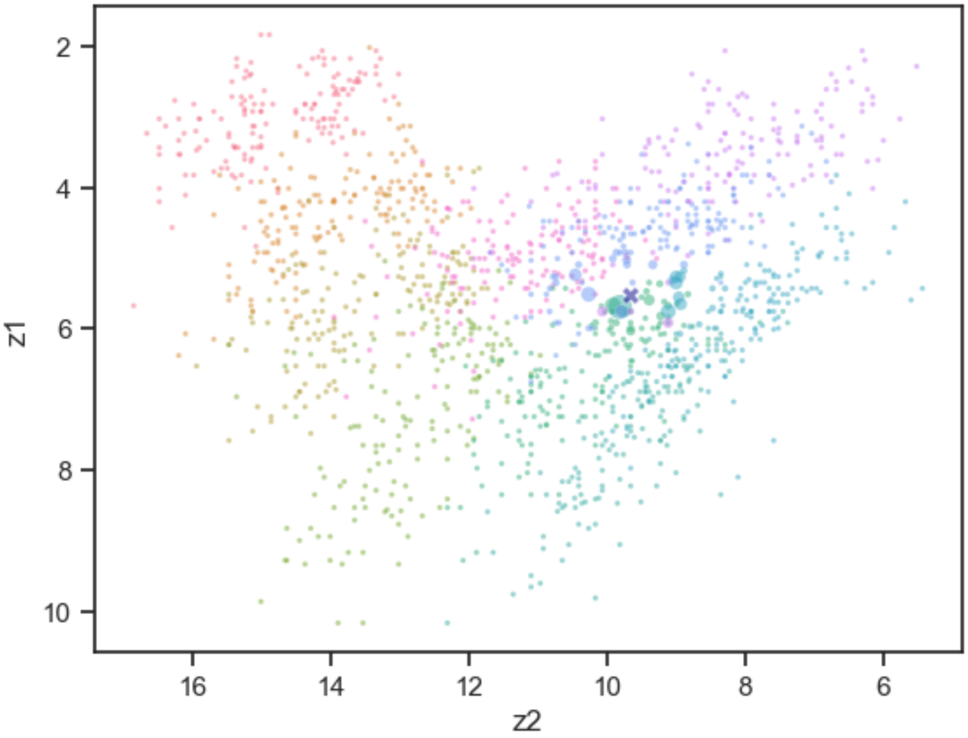
```
In [34]: # Bias towards children
         cat = "type"
         catbias = {"m":1, "c":10, "w":1}
         exemplars = exclude(exemplars, thestim,
                             exclude_self = True,
                             alsoexclude='speaker')
         exemplars = bias_N(exemplars, cat=cat,
                            catbias=catbias)
         bigdf = activation(thestim, exemplars,
                            dimsdict = w2, c = cval)
         pr = probs(bigdf, catslist)
         pr['vowel']=np.round(pr['vowel'],3)
         a = getactiv(activxn = bigdf, x='z2', y = 'z1',
                      cat='vowel')
         pl = activplot(a=a, x='z2', y='z1', cat='vowel',
                        test=thestim, invert = True)
         plt.show
         #print(pr)

Out[34]: <function matplotlib.pyplot.show(close=None, block=None)>
```



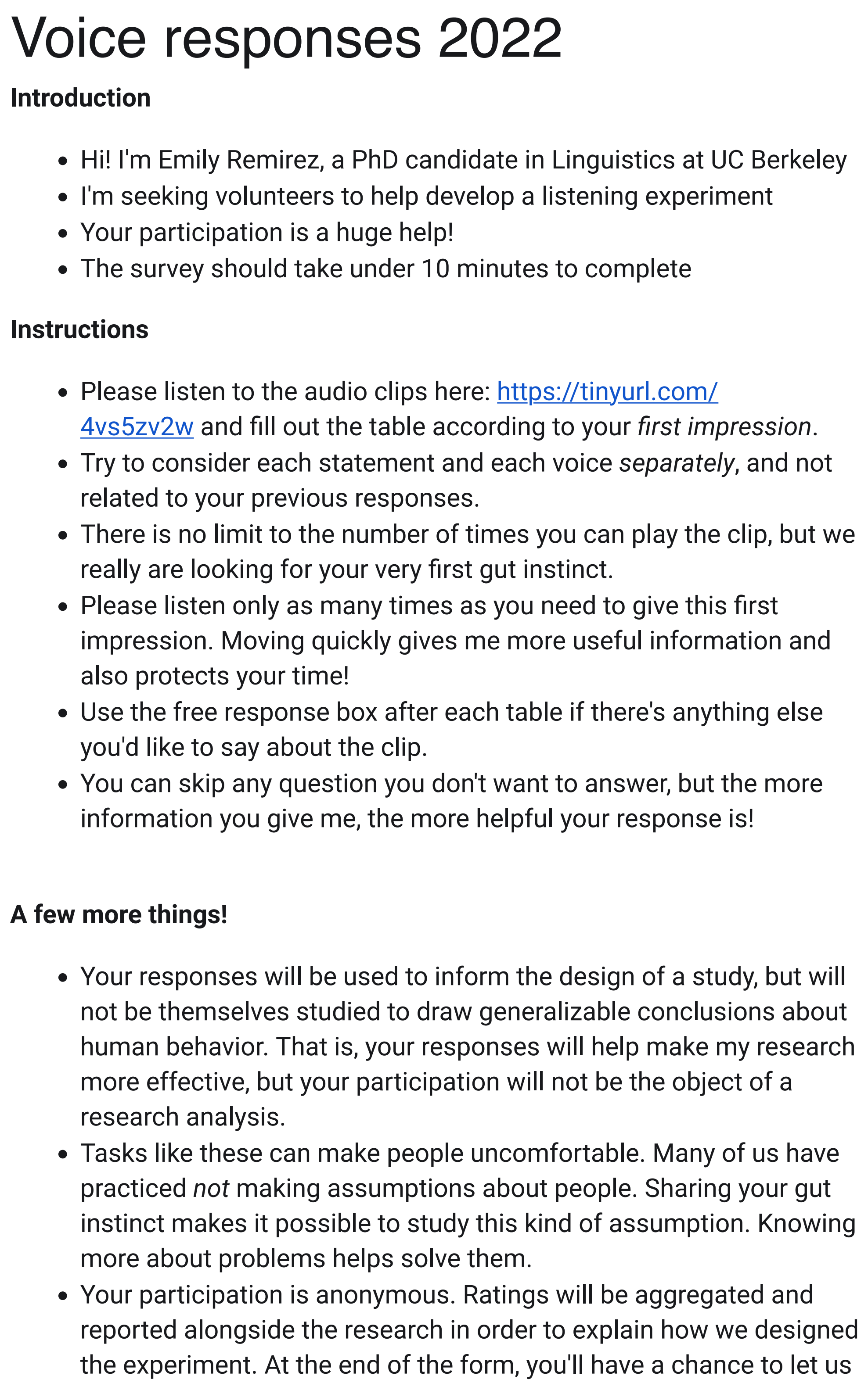
In [29]:

```
rescat = "type"

exemplars = exclude(exemplars, thestim,
                    exclude_self = True,
                    alsoexclude='speaker')
exemplars = reset_N(exemplars, N=1)
bigdf = activation(thestim, exemplars,
                   dimsdict = w2, c = cval)
pr = probs(bigdf, catslist)

#resonate!!
for n in range(0,1):
    edict = pr[rescat].set_index(rescat).to_dict()['probabili
    exemplars['resterm'] = exemplars[rescat].map(edict) / (n+
    exemplars['N'] = exemplars['N'] + exemplars['resterm']
    bigdf = activation(thestim, exemplars, dimsdict = w2, c =
    pr = probs(bigdf, catslist)

pr['vowel']=np.round(pr['vowel'],3)
a = getactiv(activxn = bigdf, x='z2', y = 'z1', cat='vowel')
pl = activplot(a=a, x='z2', y='z1', cat='vowel',
               test=thestim, invert = True)
plt.show
print(pr)
```

```
{'type':   type  probability
0    c     0.288510
1    m     0.233064
2    w     0.478427, 'vowel':      vowel  probability
0    DRESS        0.000
1   FLEECE        0.000
2     FOOT        0.277
3    GOOSE        0.056
4      KIT        0.000
5    NURSE        0.000
6     PALM        0.150
7    STRUT        0.249
8  THOUGHT        0.268
9     TRAP        0.000}
```



Appendix 3: Norming questionnaire



Voice responses 2022

Introduction

- Hi! I'm Emily Remirez, a PhD candidate in Linguistics at UC Berkeley
- I'm seeking volunteers to help develop a listening experiment
- Your participation is a huge help!
- The survey should take under 10 minutes to complete

Instructions

- Please listen to the audio clips here: https://tinyurl.com/4vs5zv2w and fill out the table according to your *first impression*.
- Try to consider each statement and each voice *separately*, and not related to your previous responses.
- There is no limit to the number of times you can play the clip, but we really are looking for your very first gut instinct.
- Please listen only as many times as you need to give this first impression. Moving quickly gives me more useful information and also protects your time!
- Use the free response box after each table if there's anything else you'd like to say about the clip.
- You can skip any question you don't want to answer, but the more information you give me, the more helpful your response is!

A few more things!

- Your responses will be used to inform the design of a study, but will not be themselves studied to draw generalizable conclusions about human behavior. That is, your responses will help make my research more effective, but your participation will not be the object of a research analysis.
- Tasks like these can make people uncomfortable. Many of us have practiced *not* making assumptions about people. Sharing your gut instinct makes it possible to study this kind of assumption. Knowing more about problems helps solve them.
- Your participation is anonymous. Ratings will be aggregated and reported alongside the research in order to explain how we designed the experiment. At the end of the form, you'll have a chance to let us

know whether it's okay to reproduce any of your free response comments.

Sections

1. Audio clips A-D
2. Audio clips E-H
3. A bit about you
4. Wrap-up

Audio clips 1/2: A-D

The audio files are linked in this Google Slides presentation: https://tinyurl.com/4vs5zv2w; you can also access the folder here: https://tinyurl.com/ydwpdtb8

(To keep your participation anonymous, you may wish to access the files in a private browsing window or while signed out of any google accounts!)

1. A

Mark only one oval per row.

	1 - Not at all likely to be true	2 - More likely to be false than true	3 - More likely to be true than false	4 - Very likely to be true	Decline to answer
This person has been described as an influencer	○	○	○	○	○
This person has been described as "chill"	○	○	○	○	○
This person surfs and/or skates	○	○	○	○	○
I would be friends with this person	○	○	○	○	○
This person has been described as a "Valley Girl"	○	○	○	○	○
This person is a millennial / gen Y	○	○	○	○	○

This person is gen Z	○	○	○	○	○
This person has been described as a "business professional"	○	○	○	○	○
This person is from California	○	○	○	○	○
This person has been described as "White"	○	○	○	○	○
This is a real person and not a robot	○	○	○	○	○
This person is from the Midwest	○	○	○	○	○

2. Comments about clip A

A bit about you

This information will help me contextualize responses. Often, the way we would respond to questions like those above depends on our own history. Knowing a bit about you will help me identify these patterns that may be relevant for my study.

17. Think about the places you consider "home" in some way.
For me, this includes: the cities in Texas where I was raised and went to college, and Berkeley, CA, where I've lived for about 7 years. It doesn't include the city in Michigan where I lived for 3 months.

Which term(s) describe those places?
I would check: SF Bay Area, California, The West Coast, and The South

Check all that apply.

□ Southern California / SoCal
□ SF Bay Area
□ California
□ "The West Coast"
□ "The East Coast"
□ "The South"
□ "The Midwest"
□ "The Northeast"
□ Another part of the USA (as you define it!)
□ Outside of the USA
□ A place where it's *not* common to engage with (social) media from the USA

18. Would you describe yourself as "online" or participating in "Internet culture"?

Mark only one oval.

○ Yes

○ No

○ Unsure

○ Other: ______________________

19. Which of the following statements about the term "Valley Girl" would you more or less agree with?

Check all that apply.

□ Someone has used it to describe me

□ I would use it to describe myself

□ The "girl" part of the term doesn't resonate with me, but the "valley" part does

□ It has mostly positive or neutral connotations for the majority of people similar to me

□ It has mostly positive or neutral connotations for the majority of people different from me

□ It's mostly used tongue-in-cheek or ironically

□ It isn't used very commonly

□ I have never heard it before in my life!

□ Other: ______________________

20. Do you think certain people use this term more than others? Do you have any thoughts about who uses it?

Many thanks!!

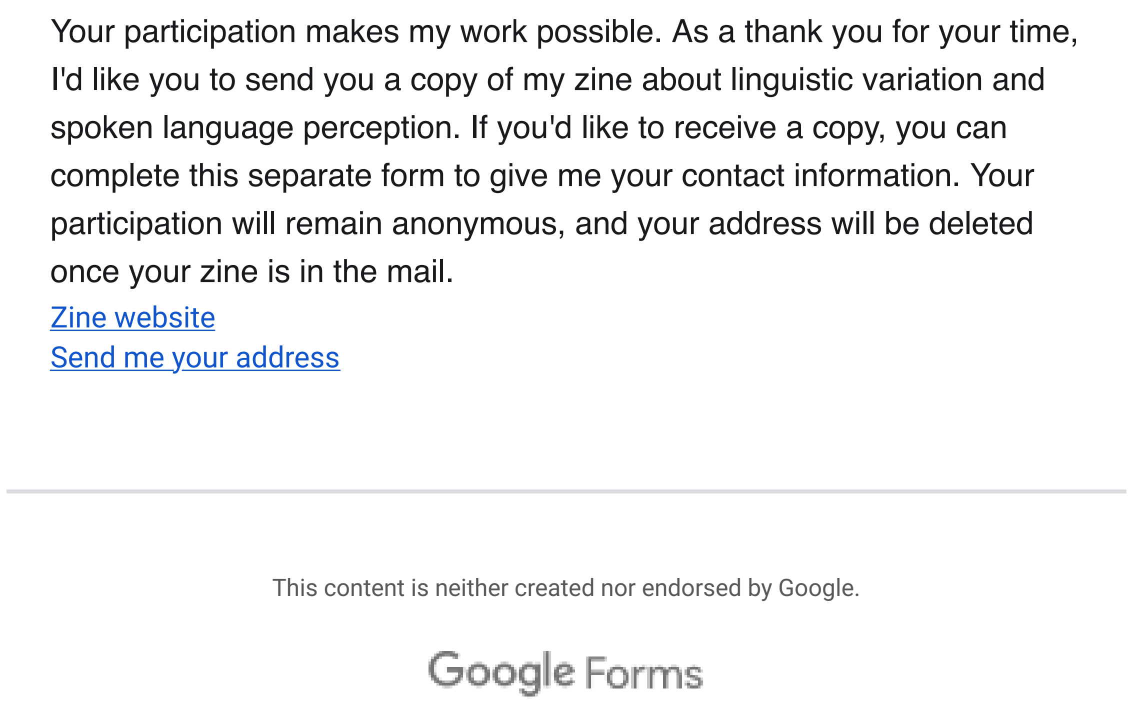
21. Is it okay to quote your comments in order to explain the design of the experiment? (Your participation will remain anonymous.)

Mark only one oval.

○ Sure, you can reprint my comments!

○ No, please don't reproduce any of my comments!

22. Any final thoughts?



Your participation makes my work possible. As a thank you for your time, I'd like you to send you a copy of my zine about linguistic variation and spoken language perception. If you'd like to receive a copy, you can complete this separate form to give me your contact information. Your participation will remain anonymous, and your address will be deleted once your zine is in the mail.

[Zine website]

[Send me your address]

This content is neither created nor endorsed by Google.