# Cognitive Modeling Assignment 3:
# a computational model of infant vowel learning

Lab sessions January 11 & 18, 2019 (Frans Adriaans)

*You are expected to write a report on the tasks in the assignment. The deadline to hand in the report is: January 26, 2019. Include your R code as a separate file.*

For this assignment you will implement a probabilistic model of early language acquisition. A general introduction to this topic was given during the lecture on Wednesday January 9, 2019. Please see the lecture slides and reading materials for additional background information.

The goal of the assignment is to model how infants learn vowel categories from acoustic input. In order to do this we need two things: (i) a data set that provides a reasonable approximation of an infant's language input, and (ii) a computational model that makes reasonable assumptions about the infant's learning mechanism. Below we will first describe the data and the basic model, and we will guide you to get everything set up in R. After taking these initial steps, you will be ready to implement the model in R, and run various simulations on the data.

## The data: American English vowels

We will assume that our "baby" (the model) is growing up in an American English environment, and hears vowels spoken by native speakers of American English. Obviously infants hear a complex speech signal which contains a lot more than just vowels, but the first simplification we will make is that the input for vowel learning consists of isolated vowel tokens. One of the most commonly cited studies on the acoustic characteristics of American English vowels is the following study:

Hillenbrand, J., Getty, L. A., Clark, M. J., & Wheeler, K. (1995). Acoustic characteristics of American English vowels. *The Journal of the Acoustical society of America*, 97(5), 3099-3111.

You can use this reference in case you are interested in the data collection process, and/or if you would like to read about extensive phonetic analyses done on the data. The data that was collected for this study is freely available from the author's webpage:

https://homepages.wmich.edu/~hillenbr/voweldata.html

Please click on *fine sampling* to download the file called 'bigdata.dat'. This file contains a total of 1668 vowel tokens (12 different vowels spoken by 139 different speakers). For each vowel token there are 29 acoustic measurements available. (The different columns are described at the top of the file.) The first thing you'll

need to do is transform this file and save it as something that can be read into R (using *read.table*). You will do this in Task 1.

---

***Task 1: Data preprocessing***

Format the data in such a way that it can be read by R. The first line of the file should be the header. The other lines should have the observations. Once you're done, use *read.table()* to load the data. There is one more remaining task in terms of data formatting: the column "filename" is composed of valuable information that we will need later. (You can read the explanation of the filename at the top of the original data file.) Use the function *extract()* provided by the *tidyr* package to split this column into multiple columns. You should end up with a large data frame that looks something like this:

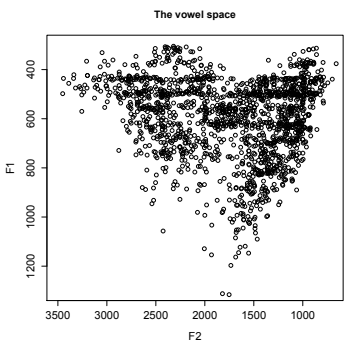
| | gender | ID | vowel | duration | F0.steady | F1.steady | … |
|---|---|---|---|---|---|---|---|
| 1 | b | 01 | ae | 257 | 238 | 630 | … |
| 2 | b | 02 | ae | 359 | 286 | 829 | … |
| … | … | … | … | … | … | … | … |

---

As discussed in class, the most important two dimensions describing a vowel are the first and second formant. The most reliable measurements for these two dimensions are given in the columns "F1.steady" (F1 at "steady state" ) and "F2.steady" (F2 at "steady state"). As with any data set, the first step should always be data exploration. What does the vowel space in this data set look like? Plotting vowel spaces can be a little bit tricky, because F2 is typically on the *x*-axis, and F1 is on the *y*-axis. To complicate things further, both axes are reversed, with values decreasing as you move further away from the origin. To save you some time, we'll provide you with a line of R code that you can use to plot the entire data set in the right space:

```
plot(d$F2.steady, d$F1.steady, xlim=rev(range(700:3500)),
ylim=rev(range(300:1300)), col="1", xlab="F2", ylab="F1", pch=1,
main="The vowel space", cex.lab=1.3, cex.axis=1.3, cex.sub=1.3)
```

This should give you the following graph:



***Task 2: A visual exploration of 'point vowels'***

We will start by focusing on a subset of the data, namely the 'point' vowels: /i/, /a/, and /u/ (which are labeled in the data as 'iy', 'ah', and 'uw'). These are the vowels that are most acoustically distinct from each other. You will find them in the corners of the acoustic space (upper left, bottom, and upper right, respectively). Make a new plot that shows only these vowels. After this first visual exploration, what is your expectation regarding the performance of a learning model trained and tested on these data? Provide the graph and a brief discussion of your expectations in your report.

---

Models are usually not trained directly on raw data, but on data sampled from a distribution that is derived from the raw data. This gives you control of the size of the training set (because you can sample as many tokens as you'd like) and allows you to manipulate the occurrence frequencies of different vowel categories (i.e., do all vowels occur equally often, or are some vowels more common than others?). Creating training and test sets involves two steps: (i) calculating the parameters of the distribution, (ii) sampling a number of tokens from the distribution. As discussed in class, the parameters in this case are each vowel category's mean and covariance matrix in F1, F2 space.

---

***Task 3: Creating training and test sets***

Calculate mean and covariance matrix (using *mean()* and *cov()*, respectively) for each of the three point vowels, and then use the function *mvrnorm()* (from the package *MASS*) to sample 2000 tokens of each category for the training set, and 2000 tokens of each category for the test set. The test set will also need to include the 'true' vowel label for each token (this will be used later for evaluation). Randomize the order of tokens in both sets. You should end up with training and test sets that look like this:

```
> head(trainingset)
         [,1]     [,2]
[1,] 356.6409 2728.224
[2,] 420.1165 3200.569
[3,] 392.6063 1238.865
[4,] 389.0279 1019.678
[5,] 465.6452 1167.512
[6,] 489.1076 1394.888
> head(testset)
         [,1]      [,2] [,3]
[1,]  491.2599 3029.7547    1
[2,]  416.8813 1432.5669    3
[3,]  386.0252 2459.5303    1
[4,]  469.1999 1337.5499    3
[5,] 1001.9380 1305.4255    2
[6,]  484.2162  784.4193    3
```

Once you have created the sets, plot your training set and compare the graph to your visual exploration of the raw data (Task 2). Include the graph in your report and briefly discuss to what extent the sampled data accurately reflects the raw data.

---

## The model: Gaussian Mixture Models (GMMs)

Now that we have prepared the training and test data we are ready to set up the learning model in R. As discussed in class, vowel categories are typically represented using multivariate Gaussian Mixture Models, where each category is defined by its mean, covariance matrix, and mixing proportion. We can build these models using the package 'mclust' (https://cran.r-project.org/web/packages/mclust/vignettes/mclust.html ). Mclust uses the EM algorithm to estimate model parameters. The following line of code will train a model (using the training set we created in Task 3) and save it to an object we will call 'model':

```
model <- Mclust(trainingset, modelName="VVV", G=3,
prior=priorControl())
```

In this case we are telling the learner to look for 3 categories (G=3) of any shape, size, and orientation (modelName="VVV"; unconstrained ellipsoid categories). The priorControl() argument is optional, but makes the algorithm a bit more efficient.

The estimated parameters can be accessed by typing 'model$parameters'. You can use these parameters to classify novel tokens (the test set). Technically this involves two steps, both of which are given below. First, the E-step of the EM-algorithm is applied to the test data, as follows:

```
predictions.prob <- estep("VVV", data=testset[,1:2],
parameters=model$parameters)
```

This gives a probabilistic category assignment, which is converted to a discrete prediction using the Maximum Likelihood criterion, as follows:

```
predictions.disc <- apply(predictions.prob$z, 1, function(x) which(x ==
max(x)))
```

You now have a vector which contains category predictions for each token in the test set:

```
> predictions.disc
  [1] 2 1 1 2 1 3 2 2 3 1 1 2 2 2 [...]
```

In order to evaluate how many of these predictions are correct we can glue the predictions to the test set, and calculate the proportion of correct responses (i.e. responses for which the predicted category matches the true category):

```
# Attach predictions to test set
testset = data.frame(cbind(testset, predictions.disc))
colnames(testset) = c("F1.steady", "F2.steady", "TrueCat", "Pred")

# Calculate proportion of correct responses
accuracy = length(testset$TrueCat[testset$TrueCat == testset$Pred]) /
length(testset$TrueCat)
```

---

***Task 4: Evaluation issues***
Train, test, and evaluate the model the model as described above. What is the accuracy of the model? Note that you might get a really high accuracy (around 0.99, which would in fact be the correct answer), but you'll most likely get a very low accuracy, which is incorrect. You will see that your model's category predictions don't line up with the true category labels. Since the clustering algorithm was trained on unlabeled data, it doesn't know which category is which. (It will just look for three different categories in the data, and assign them arbitrary numbers.) Can you figure out how to fix this and get the correct category assignments? [**Hint:** you can either plot the data to visually inspect which category is which, or you can write a clever piece of code that tries out all possible category assignments and simply picks the one with the highest accuracy.] Discuss in your report whether you managed to achieve an accuracy of 0.99, and how you managed to fix the evaluation issue discussed above.

---

If you have followed all of these steps correctly, then you now have a basic model up and running. You are now ready to explore some more interesting questions about the data and the learning model.

## Simulations

---

***Task 5: Overlapping vowels***
The 'iy'-'ah'-'uw' data was useful to develop working code, but since these categories don't overlap, it's a fairly trivial simulation, with near-100% accuracy. Set up a new simulation in which you test to what extent the overlapping vowels 'iy', 'ih' and 'eh' can be learned successfully. Include a graph of the data in
your report, and briefly discuss the results (including the classification accuracy).

---

***Task 6: Unbalanced input***

So far we have assumed that all vowels are equally common in American English. This is of course not true. It turns out that, in infant-directed speech, 'eh' occurs about half as often as 'iy' and 'ih' (Adriaans & Swingley, 2017). Run a new simulation which reflects this imbalance, and describe how this change has affected the results.

***Task 7: Three dimensions***

We might be able to obtain a better learning model if we add more acoustic dimensions. Implement a model that also uses the third formant (F3.steady), in addition to F1 and F2. To what extent does the 3-dimensional model outperform the 2-dimensional model? Do the same, but now taking duration as the third dimension. Which of these two dimensions (F3 or duration) seems to be more useful to detect vowel category structure? Discuss your findings in your report.

***Task 8: Learning from female voices***

It has been shown that infants prefer to listen to high-pitched voices, especially the mother's voice. Explore this idea by training your model on female speakers only. How does this affect the results?

***Task 9: Limitations of the model***

Obviously infants need to learn the entire vowel space, not just a handpicked set of three vowels. Explore to what extent your model can learn the entire vowel space. Can it learn all categories, or are there limits with respect to the number of categories this model can learn? Discuss your simulations and findings in your report.