# Multivariate analyses of tongue contours from ultrasound tongue imaging. Draft v0.2

Stefano Coretta[1], Georges Sakr[1]

**ARTICLE HISTORY**
Compiled April 11, 2025

[1] University of Edinburgh,

## 1. Introduction

> ⚠ Warning
>
> This is a "living" draft, meaning it is work in progress. While the code is fully functional and usable, we will be updating the textual explanation and might make minor changes to the code to improve clarity. Please, if using in research, cite the version you have consulted. The version of the draft is given in the title as "Draft vX.X" where "X" are incremental digits. See citation recommendation at the bottom of the document.

Ultrasound Tongue Imaging (UTI) is a non-invasive technique that allows researchers to image the shape of the tongue during speech at medium temporal resolution (30-100 frames per second, XXX). Typically, the midsagittal contour of the tongue is imaged, although 3D systems exist [XXX]. Recent developments in machine learning assisted image processing has enabled faster tracking of estimated points on the tongue contour (A. Wrench and Balch-Tomes 2022).

A. Wrench and Balch-Tomes (2022) have trained a DeepLabCut model to estimate and track specific flesh points on the tongue contour and anatomical landmarks as captured by UTI. The model estimates 11 "knots" from the vallecula to the tongue tip, plus three muscular-skeletal knots, the hyoid bone, the mandible base and and the mental spine where the short tendon attaches. See Figure 1 for a schematic illustration of the position of the tracked knots.

CONTACT: Stefano Coretta. Email: s.coretta@ed.ac.uk.

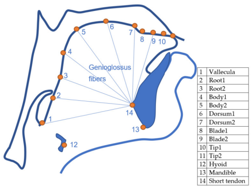


Figure 1. Schematic representation of the knots tracked by DeepLabCut. CC-BY Wrench and Balch-Tomes (A. A. Wrench 2024).

## 2. GAM

Generalised additive models (GAMs) are an extension of generalised models that allow flexible modelling of non-linear effects (Hastie and Tibshirani 1986; Wood 2006). GAMs are built upon smoothing splines functions, the components of which are multiplied by estimated coefficients to reconstruct an arbitrary time-changing curve. For a thorough introduction to GAMs we refer the reader to (Sóskuthy 2021b, 2021a; Pedersen et al. 2019; Wieling 2018).

The data tracked by DeepLabCut consists of the position on the horizontal ($x$) and vertical ($y$) axes of the fourteen knots. In this tutorial, we will focus on modelling the tongue contour based on the 11 knots from the vallecula to the tongue tip. Figure 2 illustrates the reconstructed tongue contour on the basis of the 11 knots: the shown tongue is from the offset of a vowel [o] followed by [t], uttered by a Polish speaker (see below for details on the data).

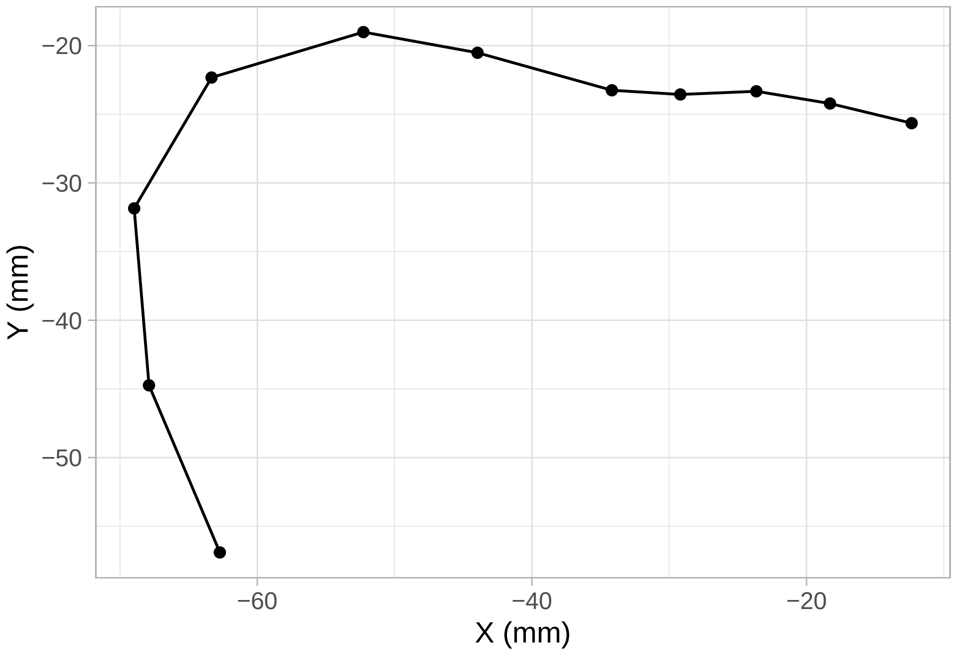


Figure 2. The eleven knots on the tongue contour taken from the offset of [o] followed by [t] (Polish speaker PL04, tongue tip to the right).

Source: Article Notebook

The same data is shown in Figure 3, but in a different format. Instead of a Cartesian coordinate system of X and Y values, the plot has knot number on the $x$-axis and X/Y coordinates on the $y$-axis. The X/Y coordinates thus form "trajectories" along the knots. These trajectories are the ones that can be modelled using GAMs and Functional Principal Component Analysis (FPCA). In this section, we will illustrate GAMs applied to the X/Y trajectories along the knots and how we can reconstruct the tongue contour from the modelled trajectories. We will use data from two case studies of coarticulation: vowel consonant (VC) coarticulation based on C place in Italian and Polish, and consonantal articulation of plain vs emphatic consonants in Lebanese Arabic.

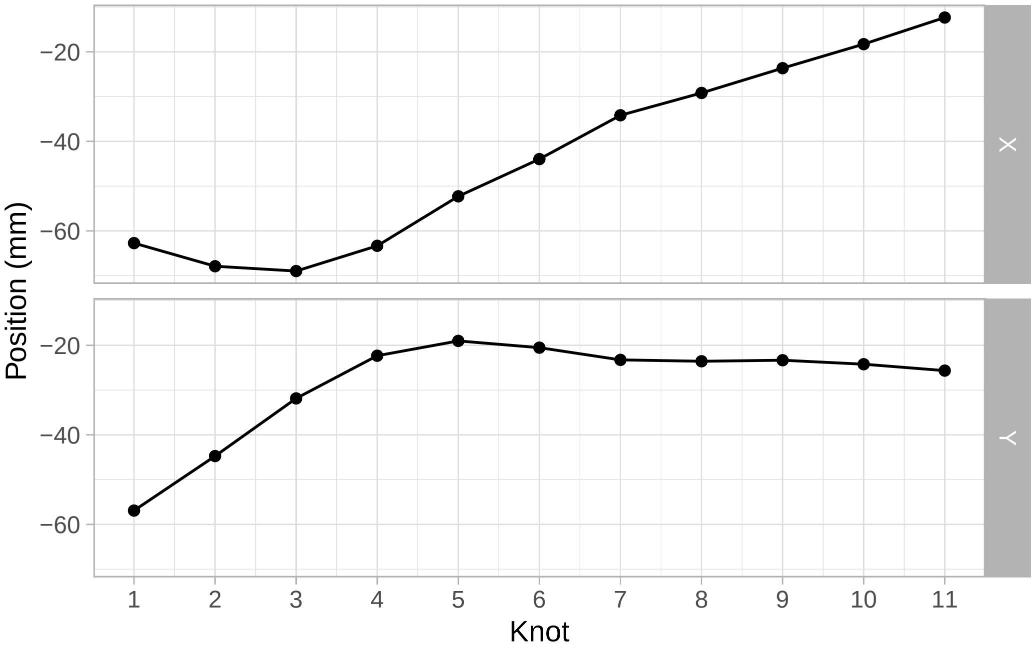


Figure 3. The horizontal and vertical positions of the elevel knots (same data as Figure 2).

Source: Article Notebook

### 2.1. *VC coarticulation*

The data of the first case study, Coretta (2018), comes from Coretta (2020b) and have been discussed in Coretta (2020a) (the analysis concerned the position of the tongue root during the duration of vowels followed by voiceless or voiced stops; in this paper we focus on tongue contours at the vowel offset). The materials are /pVCV/ words embedded in a frame sentence (*Dico X lentamente* 'I say X slowly' in Italian and *Mówię X teraz* 'I say X now' in Polish). In the /pVCV/ words, C was /t, d, k,  / and V was /a, o, u/ (in each word, the two vowels where identical, so for example *pata, poto, putu*). The data analysed here is from 9 speakers of Italian and 6 speakers of Polish (other speakers were not included due to the difficulty in processing their data with DeepLabCut).

[XXX Processing of data with DLC and filtering. Link to notebook].

The following code chunk reads the filtered data. A sample of the data is shown in Table 1. Figure 4 shows the tongue contours for each individual speaker. It is possible to notice clusters of different contours, related to each of the vowels /a, o, u/. Figure 5 zooms in on PL04 (Polish): the contours of each vowel are coloured separately, and two panels separate tongue contours taken at the offset of vowels followed by coronal (/t, d/) and velar stops (/k,  /). Crucially, the variation in tongue shape at vowel offset (or closure onset) across vowels contexts is higher in the coronal than in the velar contexts. This is not surprising, giving the greater involvement of the tongue body and dorsum (the relevant articulators of vowel production) in velar than in coronal stops.

```
dlc_voff_f <- readRDS("data/coretta2018/dlc_voff_f.rds")
```

Source: Article Notebook

Table 1. A sample of the VC coarticulation data from Coretta (2018).

| speaker | word | X | Y | knot | knot_label |
|---|---|---|---|---|---|
| it01 | pugu | -55.2105 | -44.1224 | 0 | Vallecula |
| it01 | pugu | -60.6994 | -31.3486 | 1 | Root_1 |
| it01 | pugu | -65.1434 | -17.7311 | 2 | Root_2 |
| it01 | pugu | -63.6757 | -4.2022 | 3 | Body_1 |
| it01 | pugu | -57.2505 | 7.8483 | 4 | Body_2 |
| it01 | pugu | -44.9086 | 13.3162 | 5 | Dorsum_1 |

Source: Article Notebook

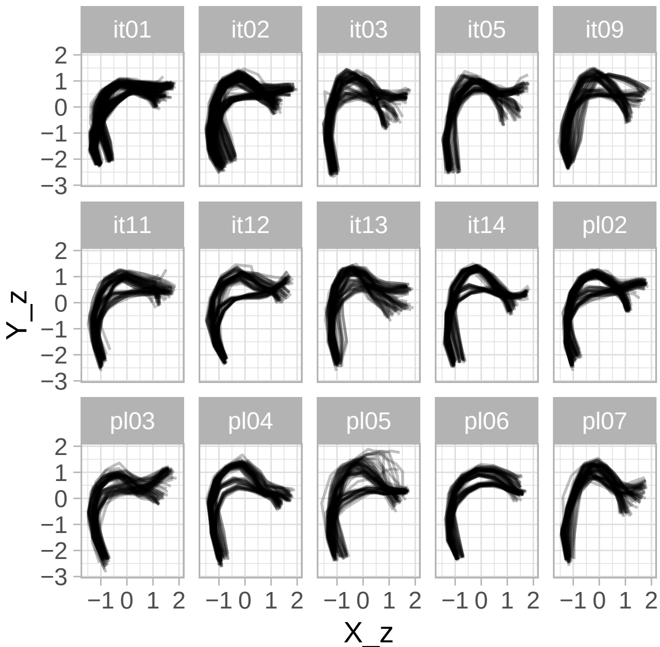


Figure 4. Tongue contours of 9 Italian speakers and6 Polish speakers, taken from the offset of the first vowel in /pCVCV/ target words.

Source: Article Notebook

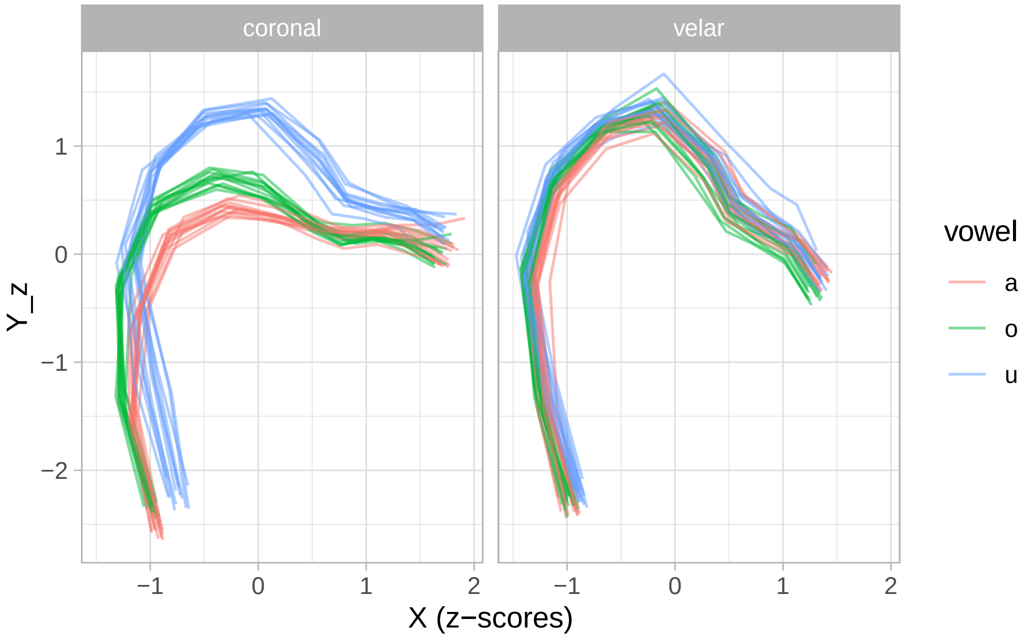


Figure 5. Tongue contours of PL04 (Polish) taken from the offset of vowels followed by coronal or velar stops. Tip is on the right.

Source: Article Notebook

We can now run a multivariate GAM to model the tongue contours. A multivariate GAM can be fitted by providing model formulae for each outcome variable (in our case, `X_z` and `Y_z`) in a list. For example `list(y ~ s(x), w ~ s(x))` would instruct `mgcv::gam()` to fit a bivariate GAM with the two outcome variables `y` and `w`. The required family is `mvn` for "multivariate normal": `mvn(d = 2)` indicates a bivariate family (a multivariate family with two dimensions, i.e. two outcome variables). In the model below, we are fitting a multivariate GAM to the $z$-scored X and Y coordinates. For both outcome variables, we include a smooth over knot (`s(knot, ...)`) with a `by` variable `vow_place_lang`: this variable is built from an interaction of vowel, place and language. (XXX note on categorical interactions in GAMs). We set `k` to 5: this will usually be sufficient for X/Y coordinates of tongue contours, since they are by nature not very "wiggly" (which would require a higher `k`). We also include a factor smooth over knot for speaker (the equivalent of a non-linear random effect) with `s(knot, speaker, ...)`: since language is a between-speaker variable, we use `vow_place` as the `by` variable (`vow_place` is the interaction of vowel and place).

```
library(mgcv)

voff_gam <- gam(
  list(
    X_z ~ vow_place_lang +
      s(knot, by = vow_place_lang, k = 5) +
      s(knot, speaker, by = vow_place, bs = "fs", m = 1),
    Y_z ~ vow_place_lang +
      s(knot, by = vow_place_lang, k = 5) +
      s(knot, speaker, by = vow_place, bs = "fs", m = 1)
```

```r
  ),
  data = dlc_voff_f,
  family = mvn(d = 2)
)
```

Source: Article Notebook

The model summary is not particular insightful. What we are normally interested in is the reconstructed tongue contours and in which locations they are similar of different across conditions. To the best of our knowledge, there isn't a straightforward way to compute sensible measures of comparison, given the multidimensional nature of the model (i.e., only one or the other outcome can be inspected at a time; moreover, difference smooths, like in Sóskuthy (2021b) and Wieling (2018), represent the difference of the *sum* of the outcome variables, rather than each outcome separately, Michele Gubian pers. comm.) We thus recommend to plot the predicted tongue contours and base any further inference on impressionistic observations on such predicted contours. Alas, there is also no straightforward way to plot predicted tongue contours, but to extract the predictions following a step-by-step procedure, like the one illustrated in the following paragraphs.

First off, one has to create a grid of predictor values to obtain predictions for. We do this with `expand_grid()` in the following code chunk. We start with unique values of `speaker`, `vow_place` and `knot` (rather than just using integers for the knots, we predict along increments of 0.1 from 0 to 10 for a more refined tongue contour). We then create the required column `vow_place_lang` by appending the language name based on the speaker ID. Note that all variables included as predictors in the model must be included in the prediction grid.

```r
# Create a grid of values to predict for
frame_voff <- expand_grid(
  # All the speakers
  speaker = unique(dlc_voff_f$speaker),
  # All vowel/place combinations
  vow_place = unique(dlc_voff_f$vow_place),
  # Knots from 0 to 10 by increments of 0.1
  # This gives us greater resolution along the tongue contour than just using 10 knots
  knot = seq(0, 10, by = 0.1)
) |>
  mutate(
    vow_place_lang = case_when(
      str_detect(speaker, "it") ~ paste0(vow_place, ".Italian"),
      str_detect(speaker, "pl") ~ paste0(vow_place, ".Polish")
    )
  )
```

Source: Article Notebook

With the prediction grid `frame_voff` we can now extract predictions from the model `voff_gam` with `predict()`. This function requires the GAM model object (`voff_gam`) and the prediction grid (`frame_off`). We also obtain the standard error of the prediction which we will use to calculate Confidence Intervals in the next step. Since we have used factor smooths for speaker, we now have to manually exclude these smooths from the prediction to obtain a "population" level prediction. We do this by listing the smooths to be removed in `excl`: note that the smooths must be named as they are in the summary of the model, so always check the summary to ensure you list all of the factor smooths. Finally, we rename the columns with the name of the outcome variables.

```
# List of factor smooths, to be excluded from prediction
excl <- c(
  "s(knot,speaker):vow_placea.coronal",
  "s(knot,speaker):vow_placeo.coronal",
  "s(knot,speaker):vow_placeu.coronal",
  "s(knot,speaker):vow_placea.velar",
  "s(knot,speaker):vow_placeo.velar",
  "s(knot,speaker):vow_placeu.velar",
  "s.1(knot,speaker):vow_placea.coronal",
  "s.1(knot,speaker):vow_placeo.coronal",
  "s.1(knot,speaker):vow_placeu.coronal",
  "s.1(knot,speaker):vow_placea.velar",
  "s.1(knot,speaker):vow_placeo.velar",
  "s.1(knot,speaker):vow_placeu.velar"
)

# Get prediction from model voff_gam
voff_gam_p <- predict(voff_gam, frame_voff, se.fit = TRUE, exclude = excl) |>
  as.data.frame() |>
  as_tibble()

# Rename columns
colnames(voff_gam_p) <- c("X", "Y", "X_se", "Y_se")
```

Source: Article Notebook

Now we have to join the prediction in `voff_gam_p` with the prediction frame, so that we have all the predictor values in the same data frame. We do so here with `bind_cols()` from the dplyr package. Note that `voff_gam_p` contains predictions for each level of the factor smooths, despite these being excluded from prediction. If you inspect the predictions for different speakers, you will find that they are the same for the same levels of `vow_place_lang`: this is because the effects of the factor smooths were removed, so `speaker` has no effect on the predicted values. This means that you can pick any Italian and Polish speaker in the predicted data frame. We do so by filtering with `filter(speaker %in% c("it01", "pl02"))`, but any other speaker would lead to the same output. We also calculate the lower and upper limits of 95% Confidence intervals (CI) for each coordinate. Note that you should interpret these CI with a grain of salt, because they are not truly multivariate, but rather represent the CI on each coordinate axis independently.

```
voff_gam_p <- bind_cols(frame_voff, voff_gam_p) |>
  # pick any Italian and Polish speaker, random effects have been removed
  filter(speaker %in% c("it01", "pl02")) |>
  # Calculate 95% CIs of X and Y
  mutate(
    X_lo = X - (1.96 * X_se),
```

```
    X_hi = X + (1.96 * X_se),
    Y_lo = Y - (1.96 * Y_se),
    Y_hi = Y + (1.96 * Y_se)
  ) |>
  # Separate column into individual variables, for plotting later
  separate(vow_place_lang, c("vowel", "place", "language"))
```

Source: Article Notebook

Figure 6 and Figure 7 show the predicted tongue contours based on the `voff_gam` model, without and with 95% CIs respectively. As mentioned earlier, there isn't a straightforward way to obtain any statistical measure of the difference between the contours on the multivariate plane, so we must be content with the figure.

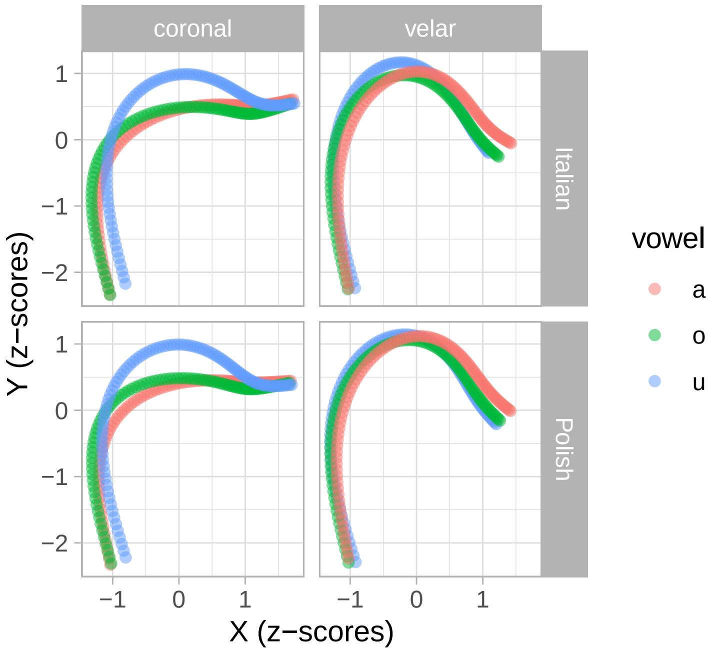


Figure 6. Predicted tongue contours based on a multivariate GAM. Uncertainty not shown.

Source: Article Notebook

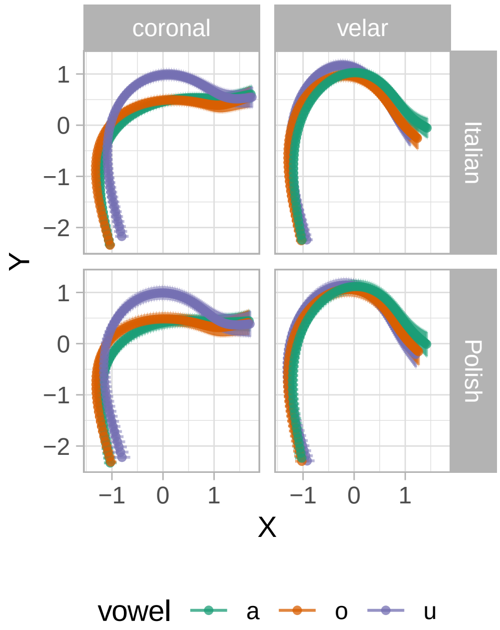


Figure 7. Predicted tongue contours based on a multivariate GAM, with 95% Confidence Intervals.

Source: Article Notebook

## 2.2. *Emphaticness*

```
dlc_emph_f <- readRDS("data/sakr2025/dlc_emph_f.rds")
```

Source: Article Notebook

Source: Article Notebook

Source: Article Notebook

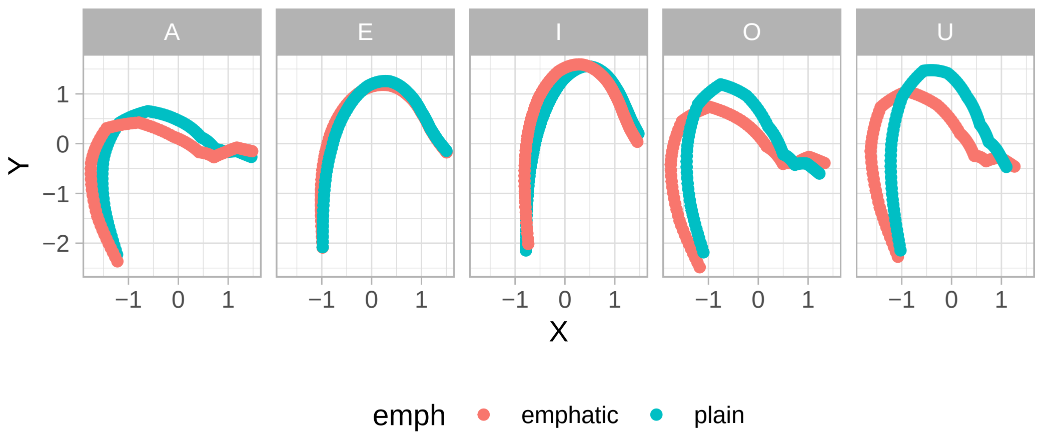


Figure 8.

Source: Article Notebook

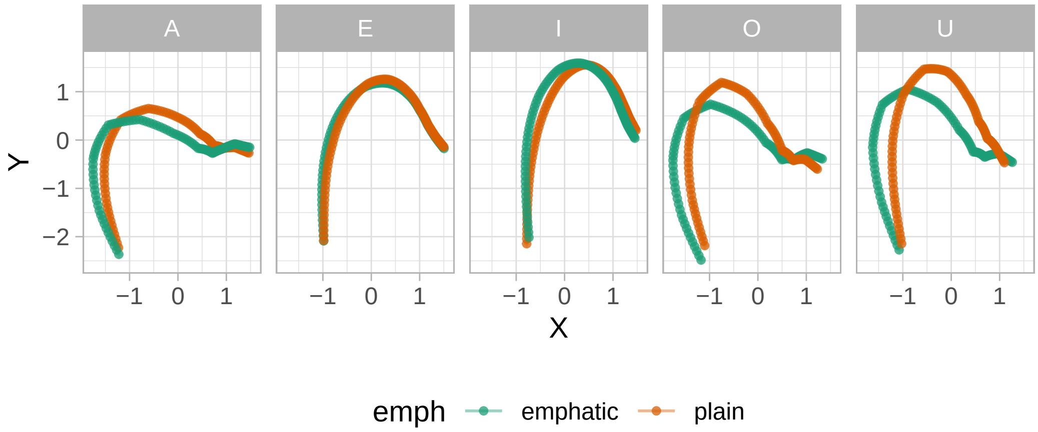


Figure 9.

Source: Article Notebook

Source: Article Notebook

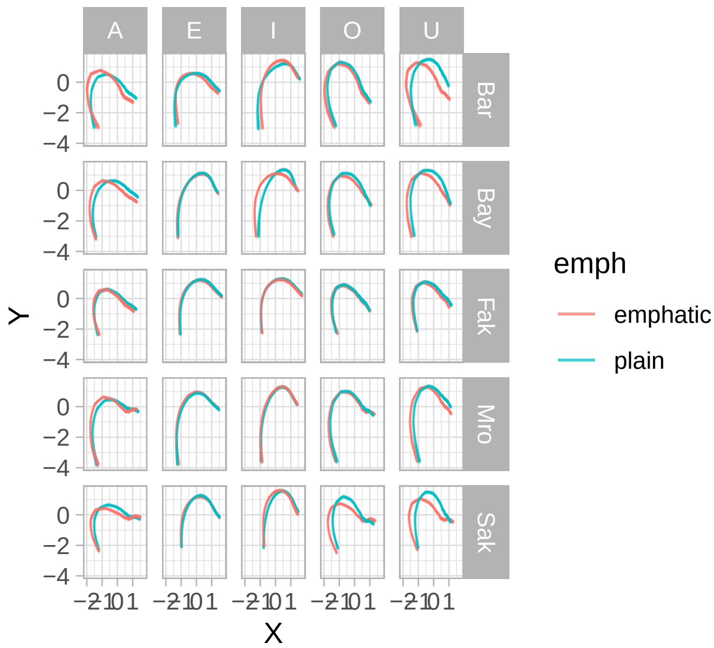


Figure 10.

Source: Article Notebook

## 3. FPCA

### 3.1. *VC coarticulation*

We will apply Multivariate Functional Principal Component Analysis (MFPCA). The following code has been adapted from Gubian (2024). The packages below are needed to run MFPCA (except landmarkregUtils, they are available on CRAN).

Source: Article Notebook

The format required to work through MFPCA is a "long" format with one column containing the coordinate labels ($x$ or $y$ coordinate) and another with the coordinate values. We can easily pivot the data with `pivot_longer()`. Note that we are using the $z$-scored coordinate values (`X_z` and `Y_z`). If you are not unsure about what the code in this section, it is always useful to inspect intermediate and final output.

Source: Article Notebook

In the second step, we create a `multiFunData` object: this is a special type of list object, with the observations of the two coordinates (`X_z` and `Y_z`) as two matrices of dimension $N \cdot 11$, where $N$ is the number of tongue contours and 11 is for the 11 knots returned by DLC. Three columns in the data are used to create the `multiFunData` object: one column with the id of each contour (in our data, `frame_id`), a time or series column (`knot`) and the column with the coordinate values (`value`).

Source: Article Notebook

Once we have our `multFunData` object, we can use the `MFPCA()` function to compute an MFPCA. In this tutorial we will compute the first two PCs, but you can compute up to $K - 1$ PCs where $K$ is the number of DLC knots in the data.

Source: Article Notebook

We can quickly calculate the proportion of explained variance of each PC with the following code. PC1 and PC2 together explain almost 100% of the variance in our data. The higher the variance explained, the better the variance patterns in the data are captured.

```
[1] 0.7108713 0.2891287
```

Source: Article Notebook

The best way to assess the effect of the PC scores on the shape of the tongue contours is to plot the predicted tongue contours based on a set of representative PC scores. In order to be able to plot the predicted contours, we need to calculate them from the MFPCA object. Gubian suggests plotting predicted curves at score intervals based on fractions of the scores standard deviation. This is what the following code does.

Source: Article Notebook

The created data frame `pc_curves` has the predicted values of the X and Y coordinates *along the knots*. This is the same structure as Figure 3, with the knot number on the $x$-axis and the coordinates on the $y$-axis. Of course, what we are after is the X/Y plot of the tongue contours, rather than the knot/coordinate plot as needed to fit an MFPCA. For the sake of clarity, we first plot the predicted curves for X and Y separately. Figure 11 shows these. The plot is composed of four panels: the top two are the predicted curves along knot number for the Y coordinates (based on PC1 in the left panel and PC2 in the right panel). Interpreting the effect of the PCs on the X and Y coordinates separately allows one to observe vertical (Y coordinate) and horizontal (X coordinate) differences in tongue position independently. However, note that the vector of muscle contractions in the tongue are not simply along a vertical/horizontal axis (Honda 1996; A. A. Wrench 2024). Looking at a full tongue contour (in an X/Y coordinates plot) will generally prove to be more straightforward.

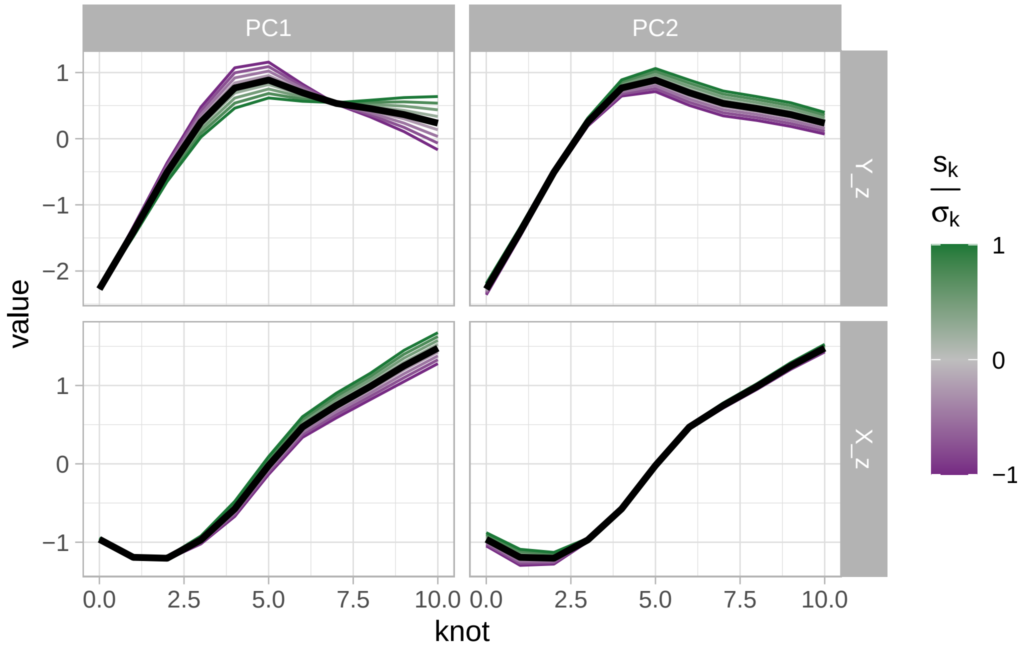


Figure 11. Predicted curves along knot number for X and Y coordinates, as obtained from an MFPCA.

Source: Article Notebook

In order to plot tongue contours in the X/Y coordinate system, we simply need to pivot the data to a wider format.

Source: Article Notebook

Figure 12 plots the predicted contours based on the the PC scores (specifically, fractions of the standard deviation of the PC scores). The $x$ and $y$-axes correspond to the X and Y coordinates of the tongue contour, with the effect of PC1 in the left panel and the effect of PC2 in the right panel. A higher PC1 score (green lines in the left panel) suggest a lowering of the tongue body/dorsum and raising of the tongue tip. Since the data contains velar and coronal consonants, we take this to be capturing the velar/coronal place of articulation effect. A higher PC2 score (green lines in the right panel) corresponds to an overall higher tongue position. Considering that the back/central vowels /a, o, u/ are included in this data set, we take PC2 to be related with the effect of vowel on the tongue shape at closure onset.

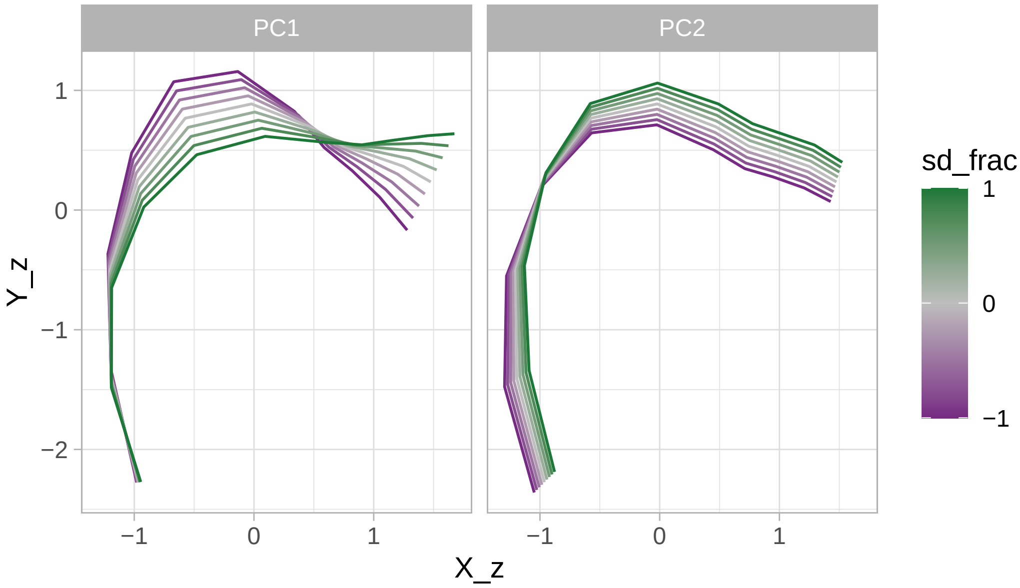

Figure 12. Predicted tongue contours as obtained from an MFPCA.

Source: Article Notebook

Given the patterns in Figure 12, we can expect to see differences in PC2 scores based on the vowel if there is VC coarticulation. We can obtain the PC scores of each observation in the data with the following code.

Source: Article Notebook

Figure 13 plots PC scores by language (rows), consonant place (columns) and vowel (colour). Both in Italian and Polish, we can observe a clear coarticulatory effect of /u/ on the production of coronal stops (and perhaps minor differences in /a/ vs /o/). On the other hand, the effect of vowel in velar stops seems to be minimal, again in both languages. This is not entirely surprising, since while coronal stops allow for adjustments of (and coarticulatory effect on) the tongue body, velar stops do not since it is precisely the tongue body/dorsum that is raised to produce the velar closure.

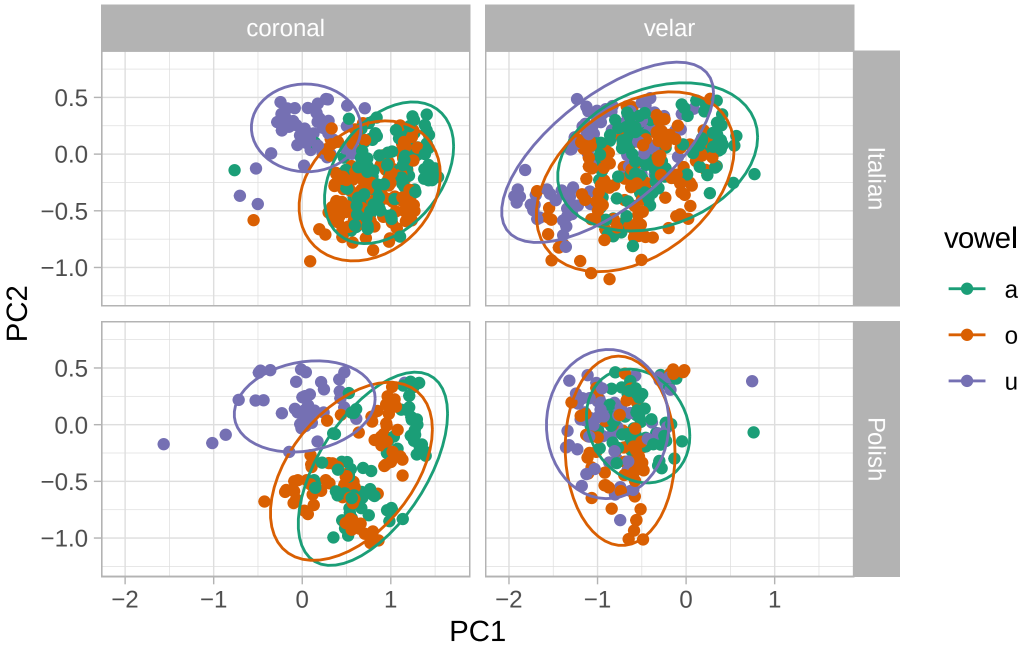

Figure 13. PC1/PC2 scores by language, consonant place of articulation and vowel.

Source: Article Notebook

Once one has established which patterns each PC is capturing, PC scores can be submitted to further statistical modelling, like for example regression models where the PC scores are outcome variables and several predictors are include to assess possible differences in PC scores.

### 3.2. *Emphaticness*

In this section we will run an MFPCA analysis on the Lebanese Arabic data. Since the procedure is the same as in the previous section, the code will not be shown here, but can be viewed in the article notebook, at XXX.

Figure 14 illustrates the reconstructed tongue contours (taken from 35 ms before the CV boundary) in Lebanese Arabic, based on the MFPCA. PC1 captures the low-back/high-front diagonal movement. PC2, on the other hand, seems to be restricted to high/low movement at the back of the oral cavity. Emphatic consonants, if produced with a constricted pharynx (i.e. paryngealised), should have a lower PC1. If on the other hand they are produced with a raised tongue dorsum (i.e. velarised), they should have a lower PC2 (lower PC scores are in purple in Figure 14).

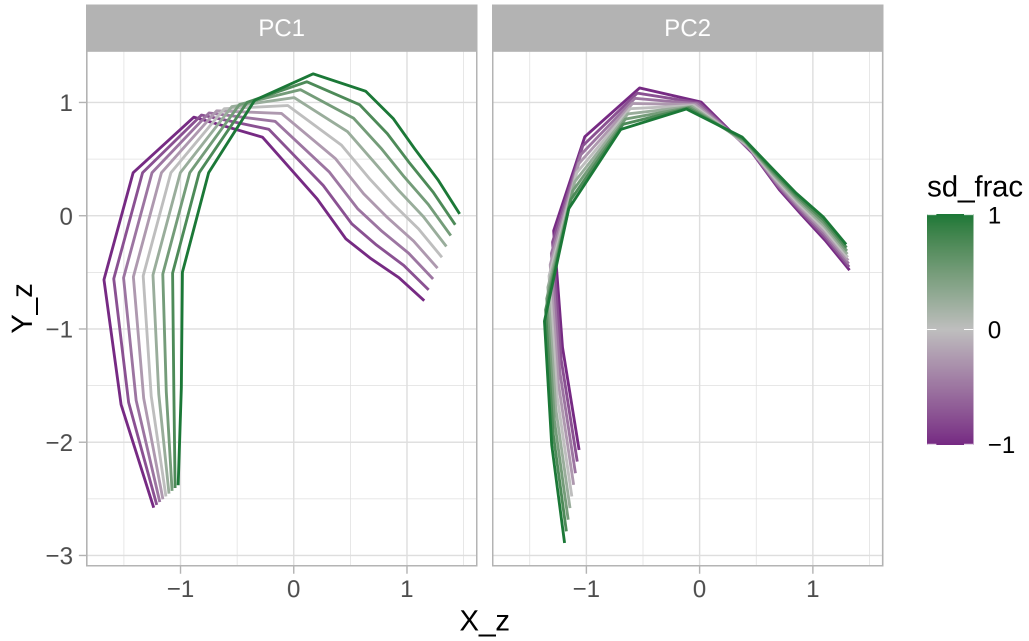

Figure 14. Predicted tongue contours of Lebanese Arabic coronal consonants as obtained from an MFPCA.

Source: Article Notebook

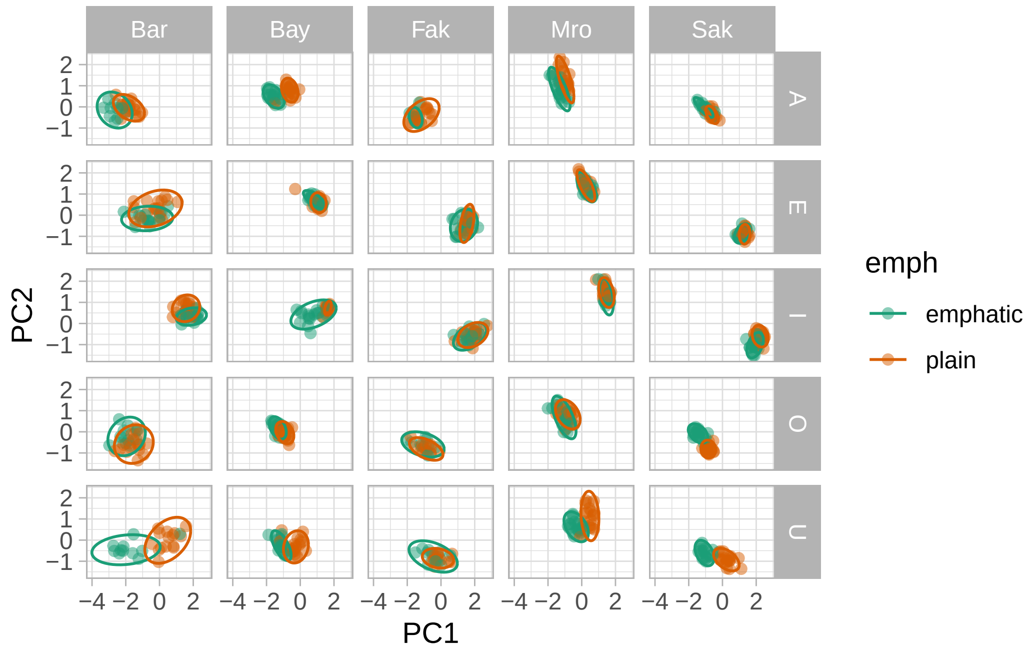

Figure 15. PC1 and PC2 scores by vowel, consonant type and speaker.

Source: Article Notebook

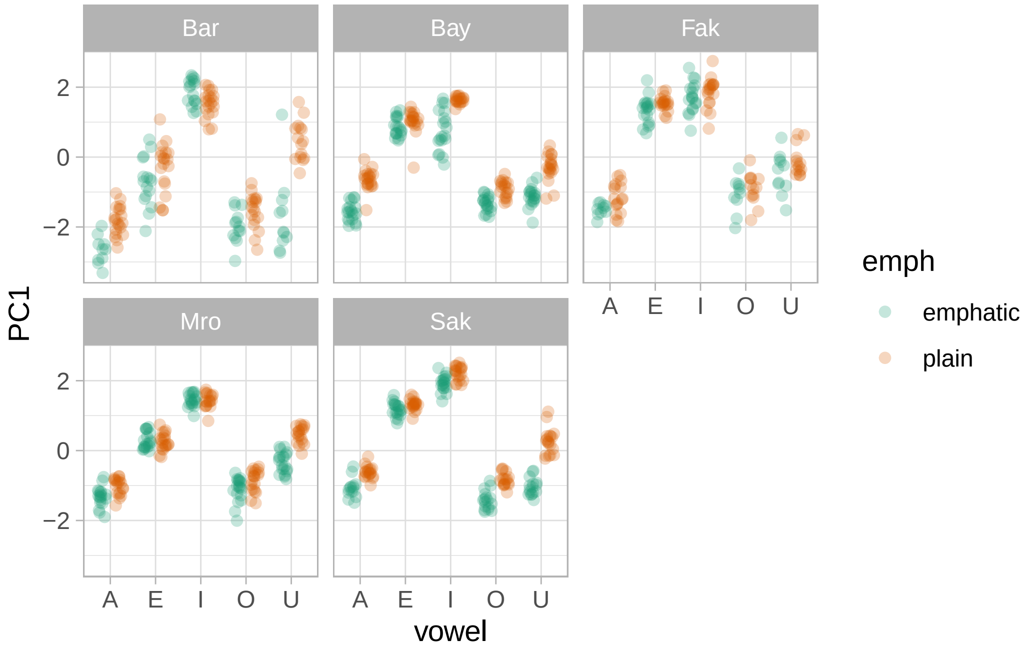


Figure 16.

Source: Article Notebook

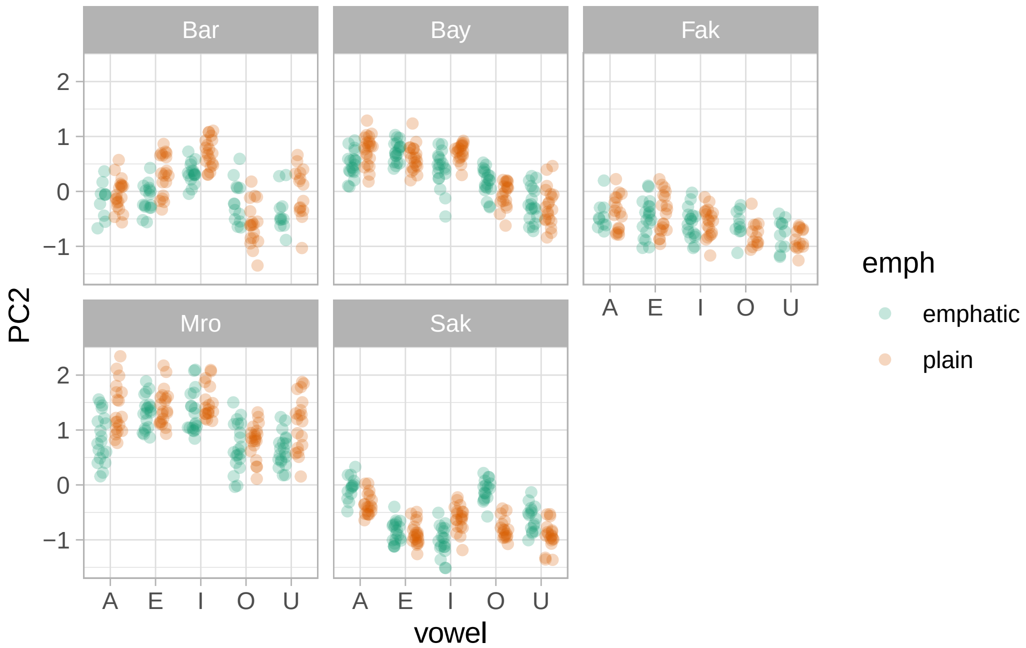


Figure 17.

Source: Article Notebook

Coretta, Stefano. 2018. "An Exploratory Study of the Voicing Effect in Italian and Polish [Data V1.0.0]." https://doi.org/10.17605/OSF.IO/8ZHKU.

———. 2020a. "Longer Vowel Duration Correlates with Greater Tongue Root Advancement at Vowel Offset: Acoustic and Articulatory Data from Italian and Polish." *The Journal of the Acoustical Society of America* 147: 245259. https://doi.org/10.1121/10.0000556.

———. 2020b. "Vowel Duration and Consonant Voicing: A Production Study." PhD thesis.

Gubian, Michele. 2024. *Workshop on Functional PCA for Phonetics and Prosody*. https://github.com/uasolo/FPCA-phonetics-workshop.

Hastie, Trevor, and Robert Tibshirani. 1986. "Generalized Additive Models." *Statistical Science* 1 (3): 297310. https://doi.org/10.1201/9780203753781-6.

Honda, Kiyoshi. 1996. "Organization of Tongue Articulation for Vowels." *Journal of Phonetics* 24 (1): 3952. https://doi.org/10.1006/jpho.1996.0004.

Pedersen, Eric J., David L. Miller, Gavin L. Simpson, and Noam Ross. 2019. "Hierarchical Generalized Additive Models in Ecology: An Introduction with Mgcv." *PeerJ* 7: e6876. https://doi.org/10.7717/peerj.6876.

Sóskuthy, Márton. 2021a. "Evaluating Generalised Additive Mixed Modelling Strategies for Dynamic Speech Analysis." *Journal of Phonetics* 84: 101017. https://doi.org/10.1016/j.wocn.2020.101017.

———. 2021b. "Generalised Additive Mixed Models for Dynamic Analysis in Linguistics: A Practical Introduction." https://doi.org/10.48550/arXiv.1703.05339.

Wieling, Martijn. 2018. "Analyzing Dynamic Phonetic Data Using Generalized Additive Mixed Modeling: A Tutorial Focusing on Articulatory Differences Between L1 and L2 Speakers of English." *Journal of Phonetics* 70: 86116. https://doi.org/10.1016/j.wocn.2018.03.002.

Wood, Simon. 2006. *Generalized Additive Models: An Introduction with r*. CRC Press.

Wrench, Alan A. 2024. "The Compartmental Tongue." *Journal of Speech, Language, and Hearing Research* 67 (10S): 38873913. https://doi.org/10.1044/2024_jslhr-23-00125.

Wrench, Alan, and Jonathan Balch-Tomes. 2022. "Beyond the Edge: Markerless Pose Estimation of Speech Articulators from Ultrasound and Camera Images Using DeepLabCut." *Sensors* 22 (3): 1133. https://doi.org/10.3390/s22031133.