Working with Vowels part 2: Plotting vowel data

Matthew Winn
University of Washington (now at University of Minnesota)

February 26, 2016

Plotting vowels using the Hillenbrand et al. (1995) data set

This is a tutorial on how to visualize vowel formant measurements using the Hillenbrand et al. (1995) data base. The script assumes that you have already created the R data objects that were produced using the "Cleaning and formatting vowel data" tutorial (http://www.mattwinn.com/tools/HB95_1.html). If you haven't downloaded that, you can just download the objects here (http://www.mattwinn.com/tools/HB95_data.RData).

All data objects in this tutorial come from that .RData file and can be examined thoroughly by loading them or viewing the first tutorial.

First Load the required packages

```
library("ggplot2")
library("dplyr")
library("scales")
library("animation")
```

Note that the animation function requires installation of ImageMagick (http://www.imagemagick.org/script/binary-releases.php).

Call up the data:

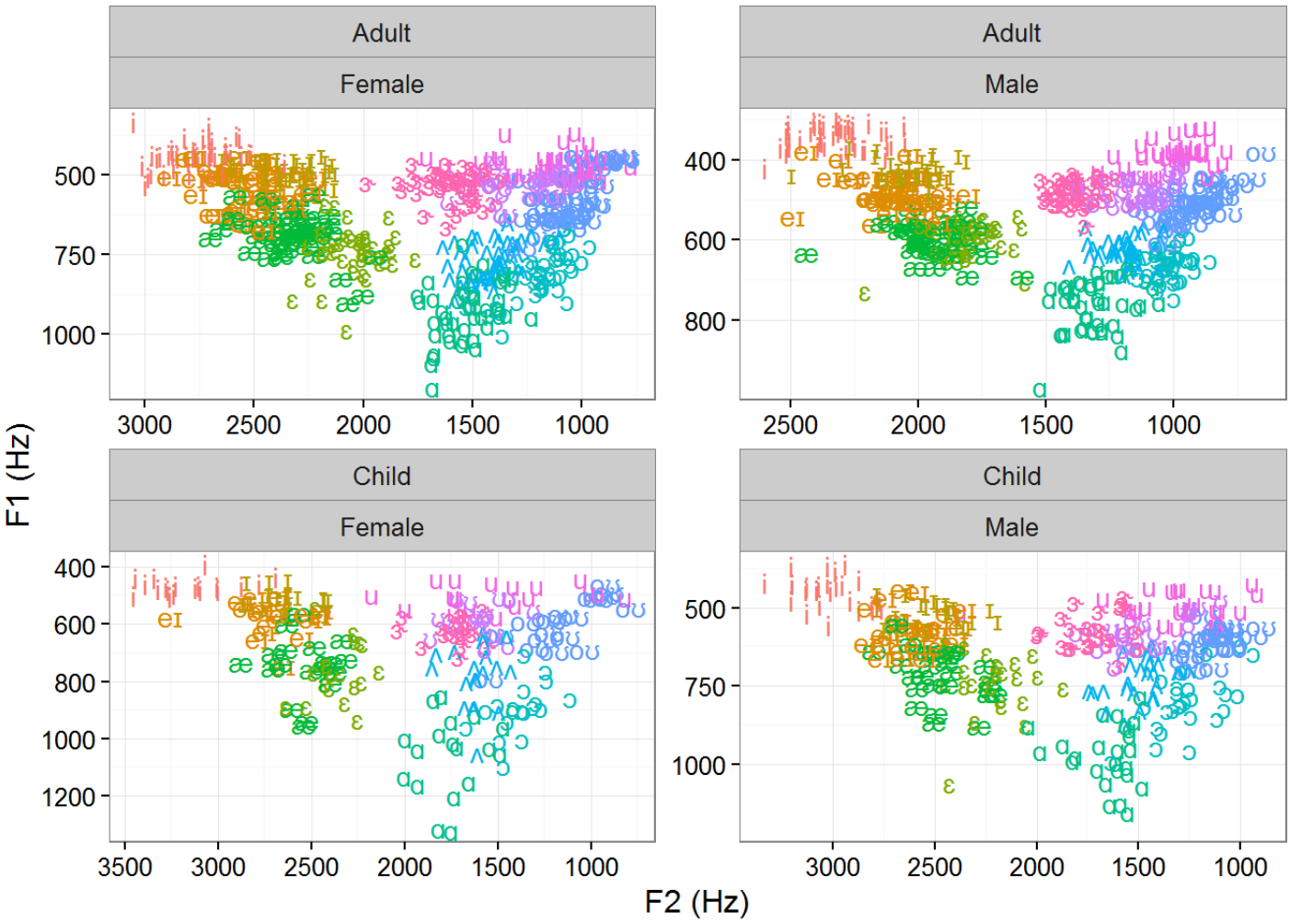
First guide R to the directory where the R objects live on your computer (i.e. where you put it from the first tutorial, or where you just downloaded the data using the link above).

```
folder_with_saved_objects <- "C:\\Enter\\Your\\Folder\\Path\\Here"
setwd(folder_with_saved_objects)
load("HB95_data.RData")
```

Plot steady-state formant values from ALL tokens

Use the `data.wide.ss` object, as it contains formant values for each of the vowels grouped by talker gender and age. The data is in “wide” format because each of the formnt values is coded as a different variable (as opposed to being a single column, whose formant number is coded in an adjacent column). Wide-format data is required when setting different formants (e.g. F1 nd F2) to separate axes.
Note how for each talker group, we let each axis limits adjust to the range of data for that group using `scales="free"`.

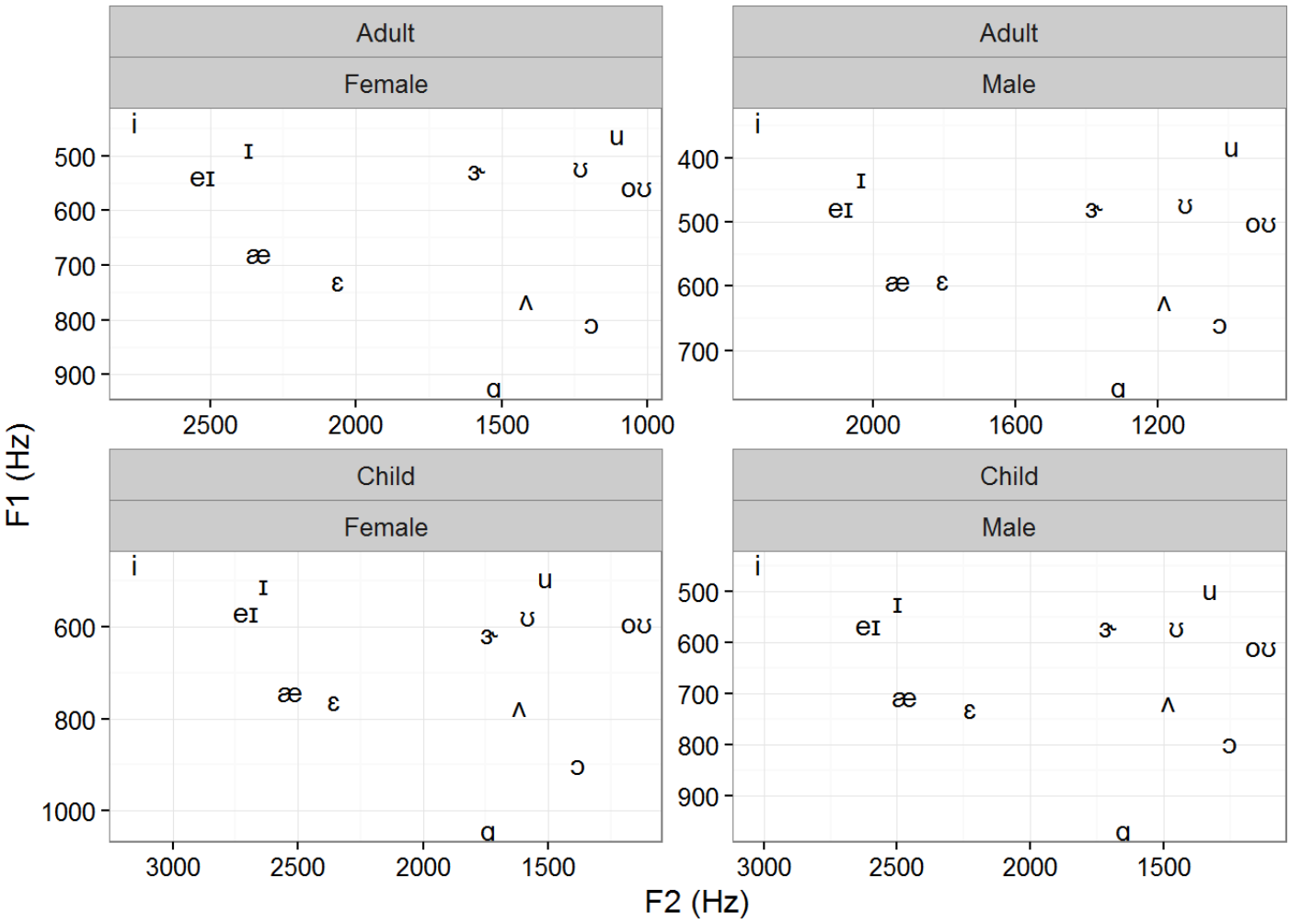
```
px_all_pts <- ggplot(data.wide.ss)+
  aes(x=f2, y=f1,
      label=Vowel.IPA,
      color=Vowel.ordered)+
  scale_x_reverse(name="F2 (Hz)")+scale_y_reverse(name="F1 (Hz)")+
  geom_text()+
  theme_bw()+
  theme(legend.position="none")+
  facet_wrap(~ Age + Gender, scales="free")
px_all_pts
```



Plot simple means for each vowel

While it is easy to produce a `mean` of the y-axis value inline in the ggplot code (e.g. with `stat_summary()`), it's not clear how to do that for the y-axis and the x-axis *at the same time*. And yet we want to keep the same plot setup that we just produced with the previous lines of code. So what we need to do is simply substitute a different data frame to the same plot using `%+%` (subbing the `data.wide.ss.sum` data frame in place of the one without ".sum") and set a new label aesthetic to remove the redundant color info.

```
px_vowel_means_4_panel <- px_all_pts %+%
  data.wide.ss.sum +
  aes(label=Vowel.IPA, color=NULL)+
  guides(color="none")
px_vowel_means_4_panel
```

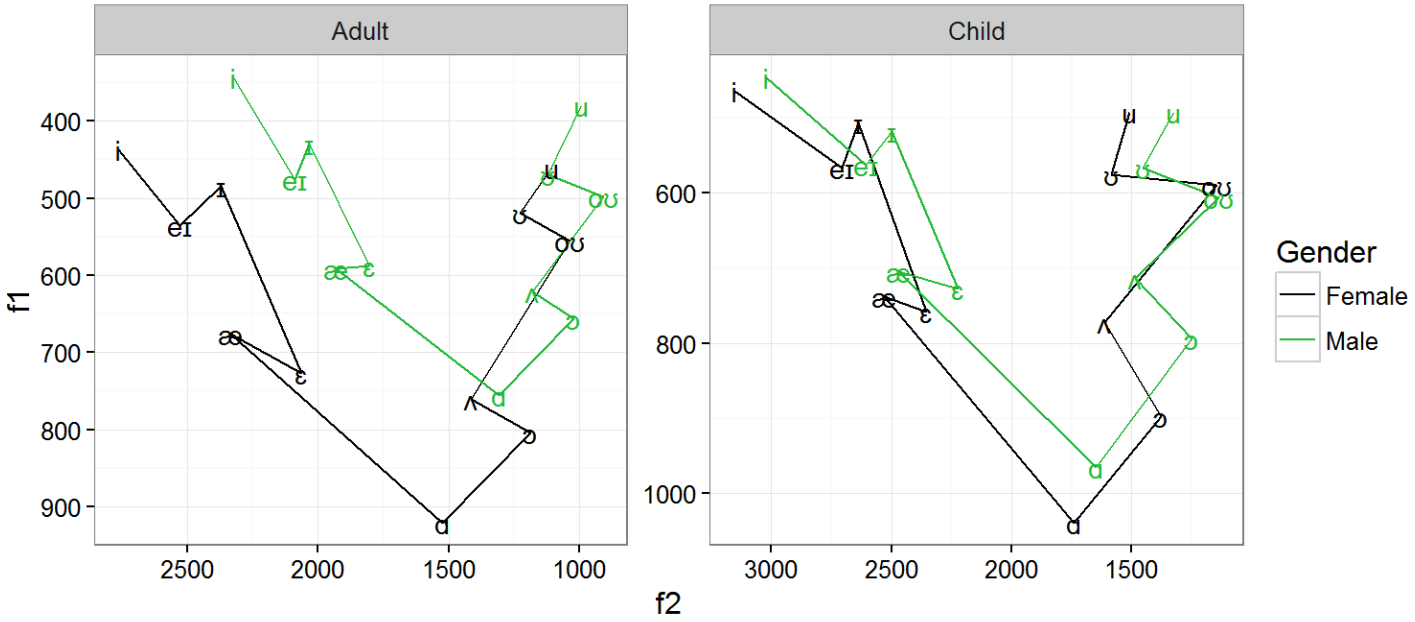


Plot Vowel space

Add a line path to define vowel space (this is why we declared a vowel order in the first script).

This code takes a data frame and first filters out the ‘er’ vowel because it’s a bit of an odd case. Then, within each group defined by Age and Gender, it arranges the data frame so that the rows are ordered by the vowel order established in the first script (it is essentially a counter-clockwise walk around the vowel space) and sends it into the `ggplot()` function.

```
px_v_space_by_gender <- data.wide.ss.sum %>%
  dplyr::filter(Vowel!="er") %>%
  group_by(Age, Gender) %>%
  arrange(as.numeric(Vowel.order.num)) %>%
  ungroup %>%
  ggplot(., aes(x=f2, y=f1, color=Gender))+
  geom_path()+
  geom_text(aes(label=Vowel.IPA), show.legend=FALSE)+
  scale_x_reverse() + scale_y_reverse()+
  theme_bw()+
  scale_color_manual(values = c("Female" = "black",
                                "Male" = "#22BA36"))+
  facet_wrap( ~ Age, scales="free")
px_v_space_by_gender
```



Convert axes to log scales

Here, a specialized funciton is needed to create a log scale in reverse:

```
reverselog_trans <- function(base = exp(1)) {
  trans <- function(x) -log(x, base)
  inv <- function(x) base^(-x)
  trans_new(paste0("reverselog-", format(base)), trans, inv,
            log_breaks(base = base),
            domain = c(1e-100, Inf))
}
```

This can be used simply as follows:

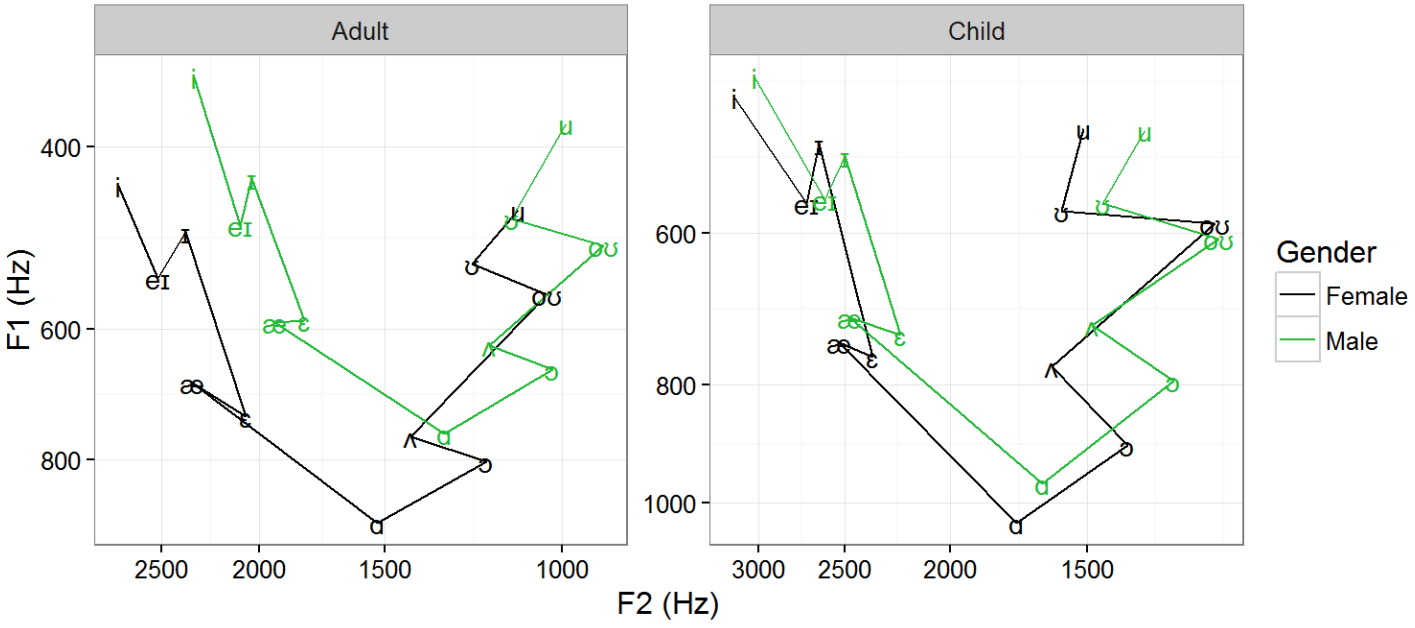
```
# p = a plot that you have already created
p_reverse_log <- p + scale_x_continuous(trans=reverselog_trans(10))
```

Now let's add it to our plot:

```
px_v_space_by_gender.log <- px_v_space_by_gender +
  scale_x_continuous(name="F2 (Hz)",
    trans=reverselog_trans(10),
                     breaks=seq(1000,3000,500))+
  scale_y_continuous(name="F1 (Hz)",
                     trans=reverselog_trans(10),
                     breaks=seq(400,1000,200))
```

```
## Scale for 'x' is already present. Adding another scale for 'x', which
## will replace the existing scale.
## Scale for 'y' is already present. Adding another scale for 'y', which
## will replace the existing scale.
```

```
px_v_space_by_gender.log
```



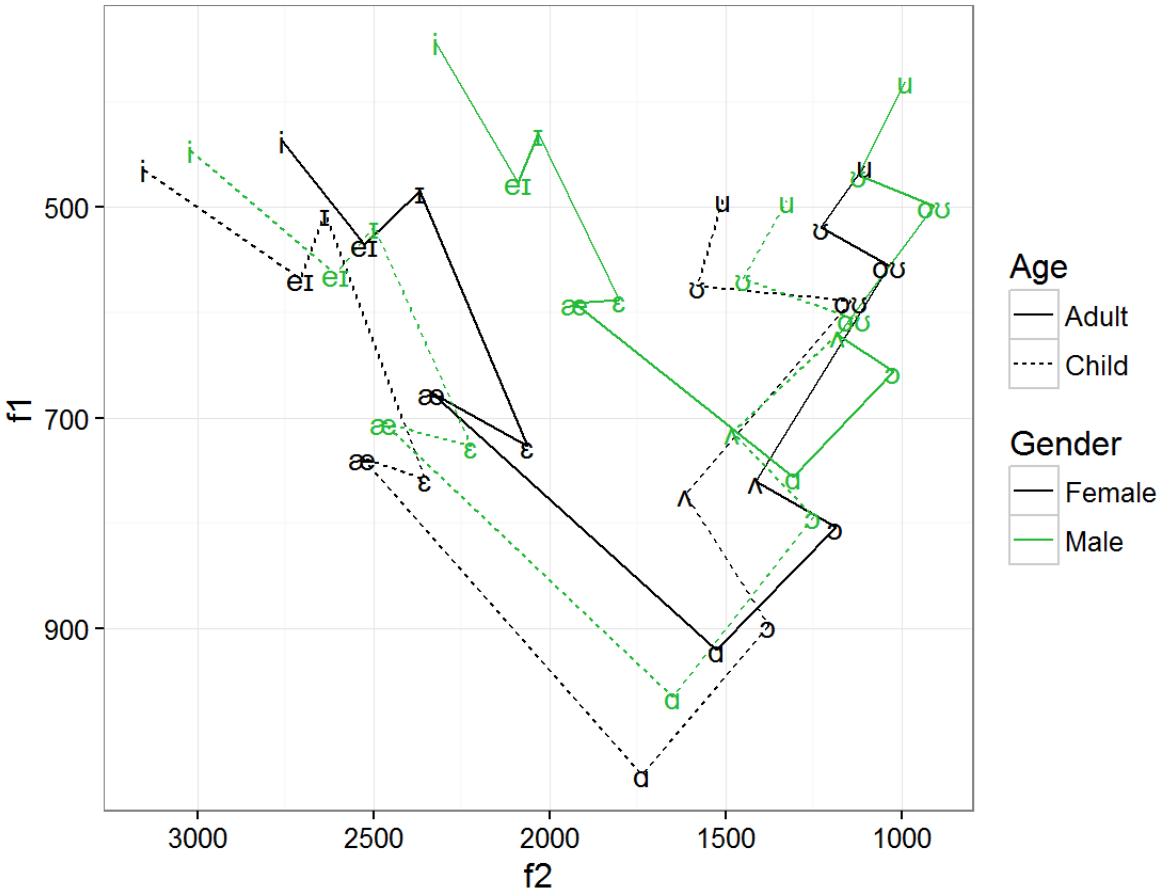
Now the data are treated in a way that more closely resembles the auditory transformation of the vowel space. There are other clever ways of doing this (using the Bark scale, ERB, or the Greenwood function).

Note fo explanatory purposes that the axis labels are **not** labeled with “log Hz” - they are labeled with linear Hz, and the spacing is simply transformed. Log-Hz values would take values between roughly 5.7 and 7.9 (for natural log) or between 2.5 and 3.4 (for base-10 log).

Plot both adults & children on the same plot

… and distinguish them by line type

```
px_v_space_by_gender_age <- px_v_space_by_gender+
  aes(linetype=Age) +
  facet_null()
px_v_space_by_gender_age
```



Plot spectrogram-style plots, showing formant tracks

Before we plot, let's make a function that makes IPA-style labels for the facets instead of the two-character codes.
Make a plain vector of the ordered vowels using the following code:

```
ordered_IPA_vowels <- data.long[,c("Vowel.IPA","Vowel.order.num")] %>%
  unique %>%
  arrange(Vowel.order.num) %>%
  `[[`(., "Vowel.IPA")
```

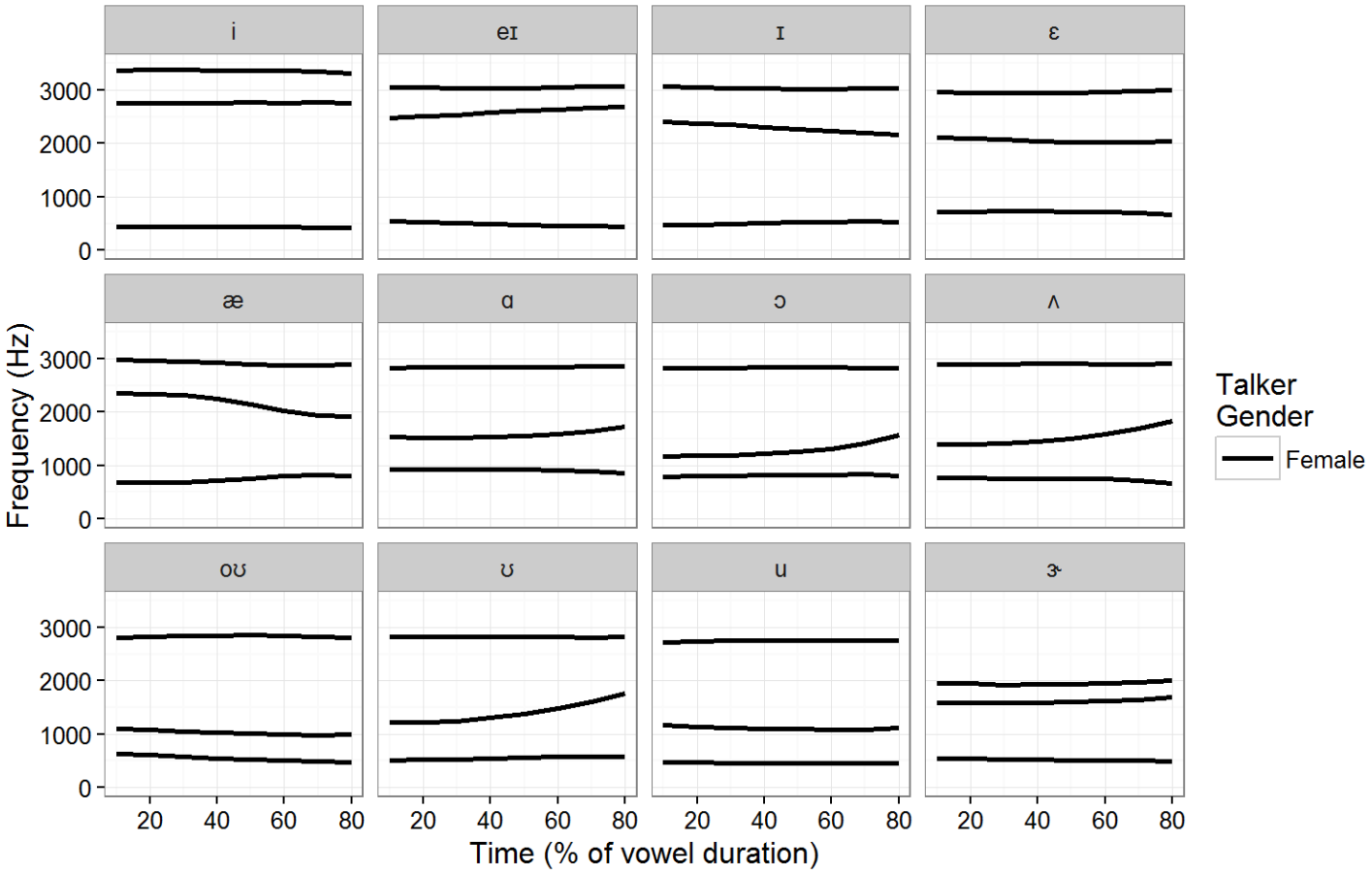
… which translates to: take the combination of vowel IPA symbols and vowel order numbers from each row in the data.long data frame, keep only unique values (no repetitions of the same value), order them according to the vowel number order column, and then extract the `Vowel.IPA` column as a single atomic vector, without any list names. (note that `[` keeps data frame list names, while `[[` is more of a pure extraction of the vector itself).

Set the order of the data.frame factor variable using that ordered vector:

```
data.long$Vowel.ordered.IPA <- factor(data.long$Vowel.IPA, levels = ordered_IPA_vowel
s)
```

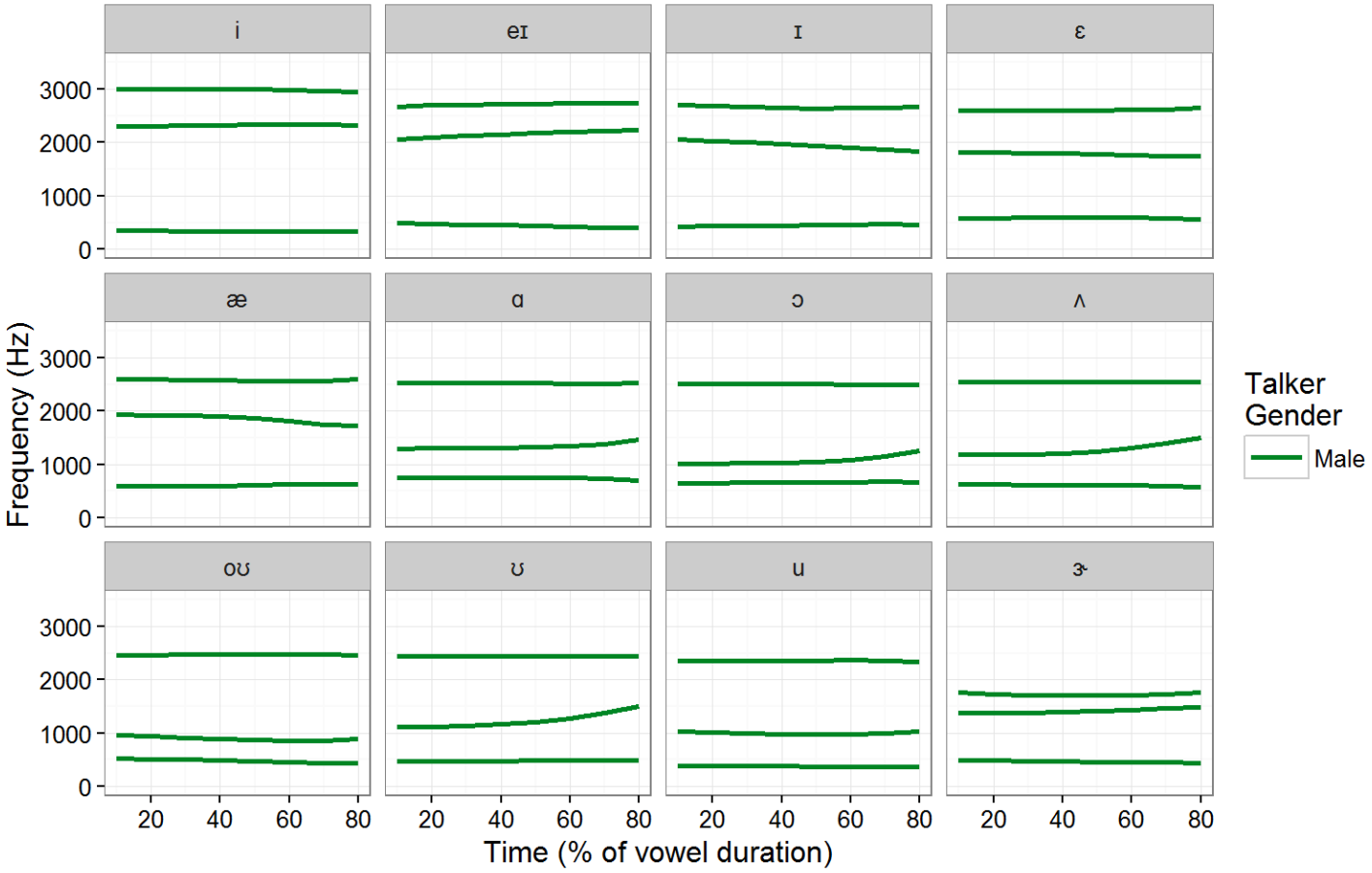
First, plot data from adult female talkers

```
px_vowel_spectro_female <- data.long %>%
  dplyr::filter(Age=="Adult", Gender=="Female") %>%
  ggplot(., aes(x=Time, y=Frequency, group=as.factor(formant)))+
  geom_line(stat="summary", fun.y="mean",
            aes(color=Gender),
            size=1.0)+
  coord_cartesian(ylim = c(0,3500))+
  scale_color_manual(name="Talker\nGender",
    values = c("Female" = "black",
               "Male" = "#008523"))+
  labs(x="Time (% of vowel duration)",
       y= "Frequency (Hz)")+
  theme_bw()+
  theme(legend.key.width = unit(1.8, "line"))+
  facet_wrap(~Vowel.ordered.IPA)
px_vowel_spectro_female
```



Then, data from male talkers:

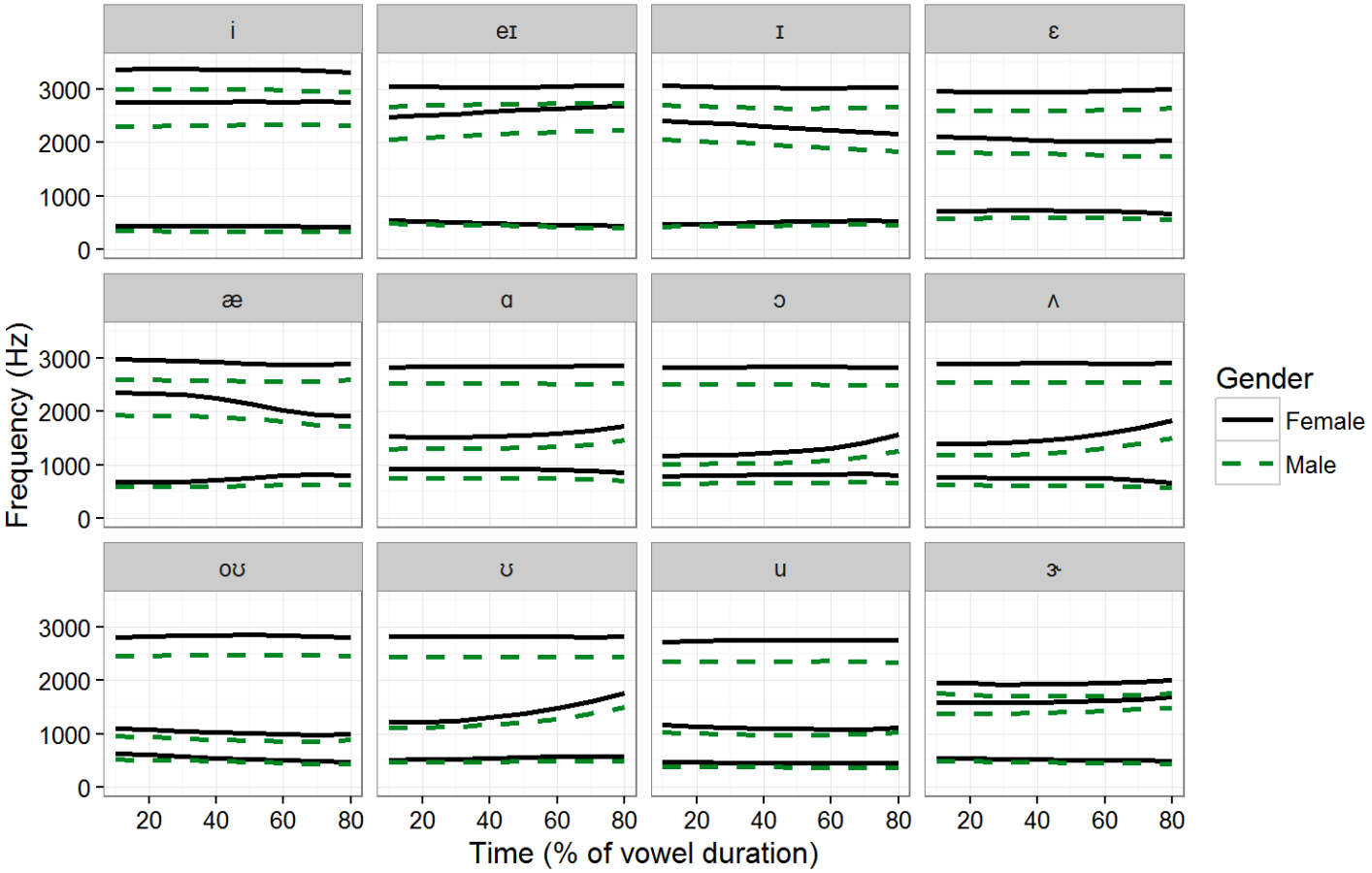
```
px_vowel_spectro_male <- px_vowel_spectro_female %+%
  (data.long %>%
     filter(Age=="Adult", Gender=="Male"))
px_vowel_spectro_male
```



The above code translates to: take the `px_vowel_spectro_female` plot, substitute in a new data frame (supplied in the parenthetical `data.long %>% dplyr::filter()` call).

Plot Women's and Men's spectrograms on the same plot:

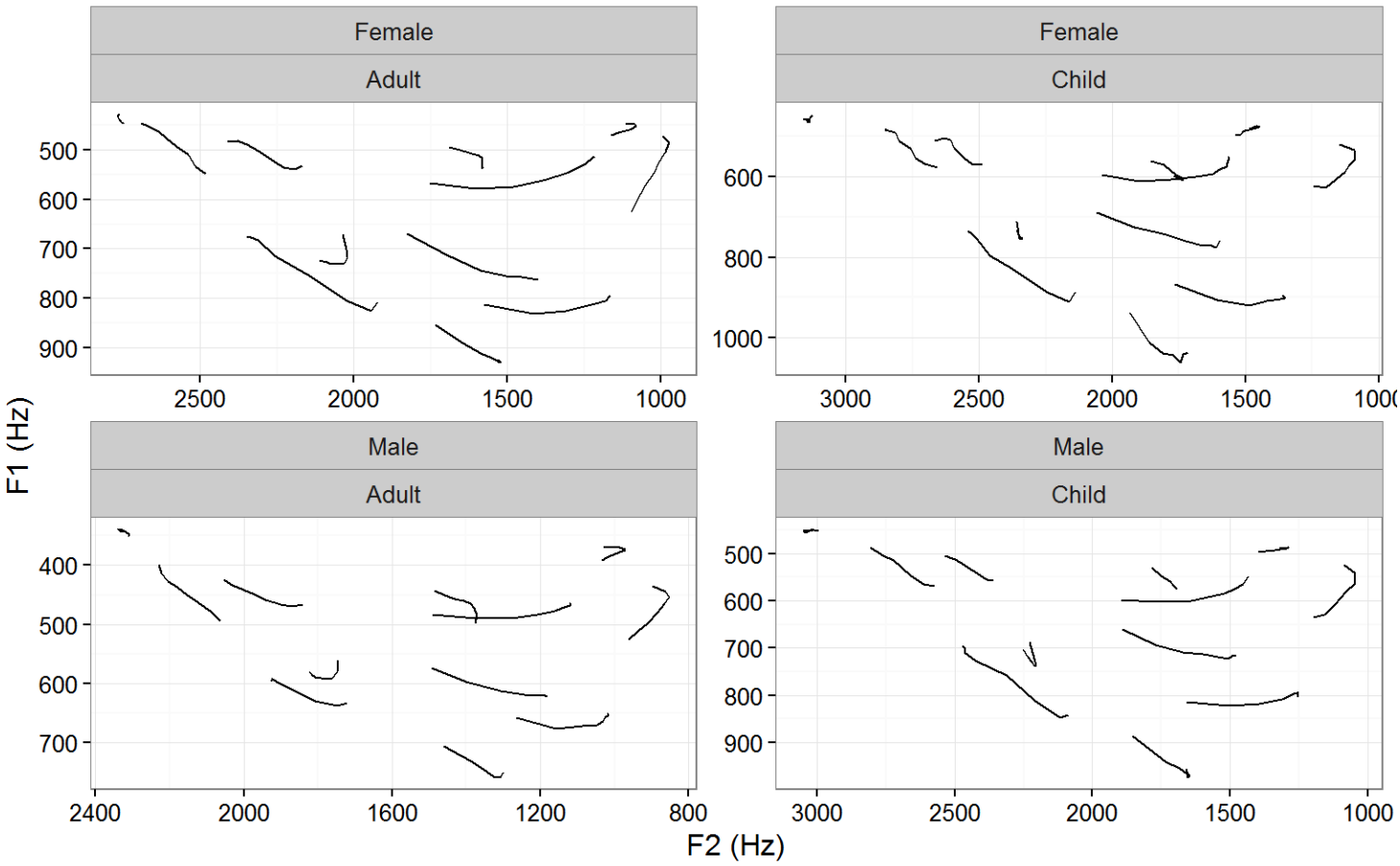
```
px_vowel_spectro_adults <- px_vowel_spectro_female %+%
  (data.long %>%
     dplyr::filter(Age=="Adult"))+
  aes(group=interaction(Gender,formant),
      color=Gender, linetype=Gender)+
  scale_color_manual(values = c("Female" = "black",
                                "Male" = "#008523"))+
  scale_linetype_manual(values = c("Female" = "solid",
                                "Male" = "dashed"))
px_vowel_spectro_adults
```



Plot articulation trajectories (VISC) within the vowel space

Here, we are plotting the vowel space again, but instead of using static values, we are using the time-series values so nicely provided by the HB95 full data set.
Vowel trajectories fall under the term “vowel-inherent spectral change”, so the shorthand “visc” is used in these plot object names.

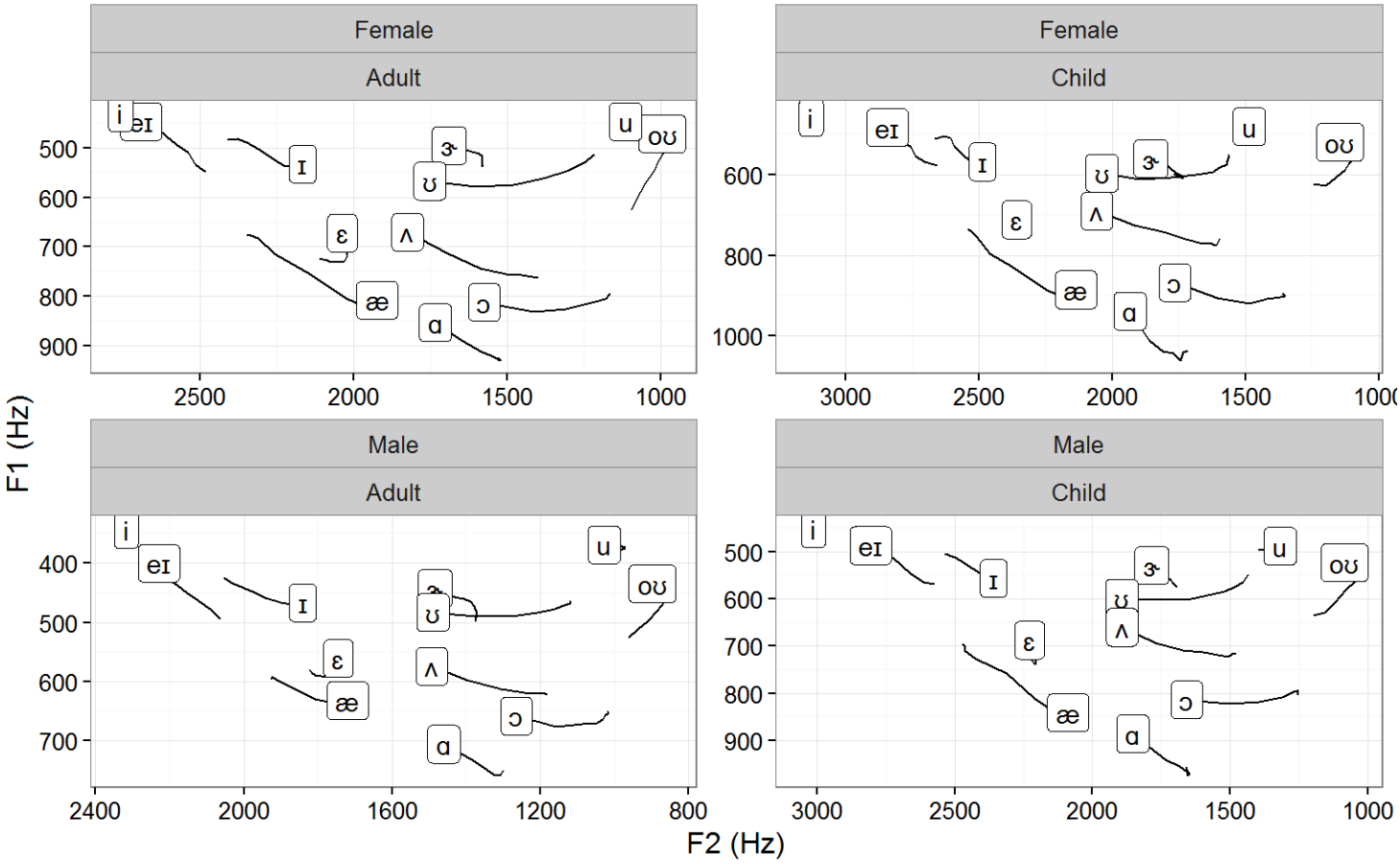
```
px_visc <- ggplot(data.wide.visc.sum) +
  aes(x=f2, y=f1, group=Vowel)+
  scale_x_reverse(name="F2 (Hz)")+scale_y_reverse(name="F1 (Hz)")+
  geom_path()+
  theme_bw()+
  facet_wrap( ~ Gender + Age, scales="free")
px_visc
```



Add vowel labels at the offset of the vowel trajectory

… using only pieces of the data frame that reflect final-time measures

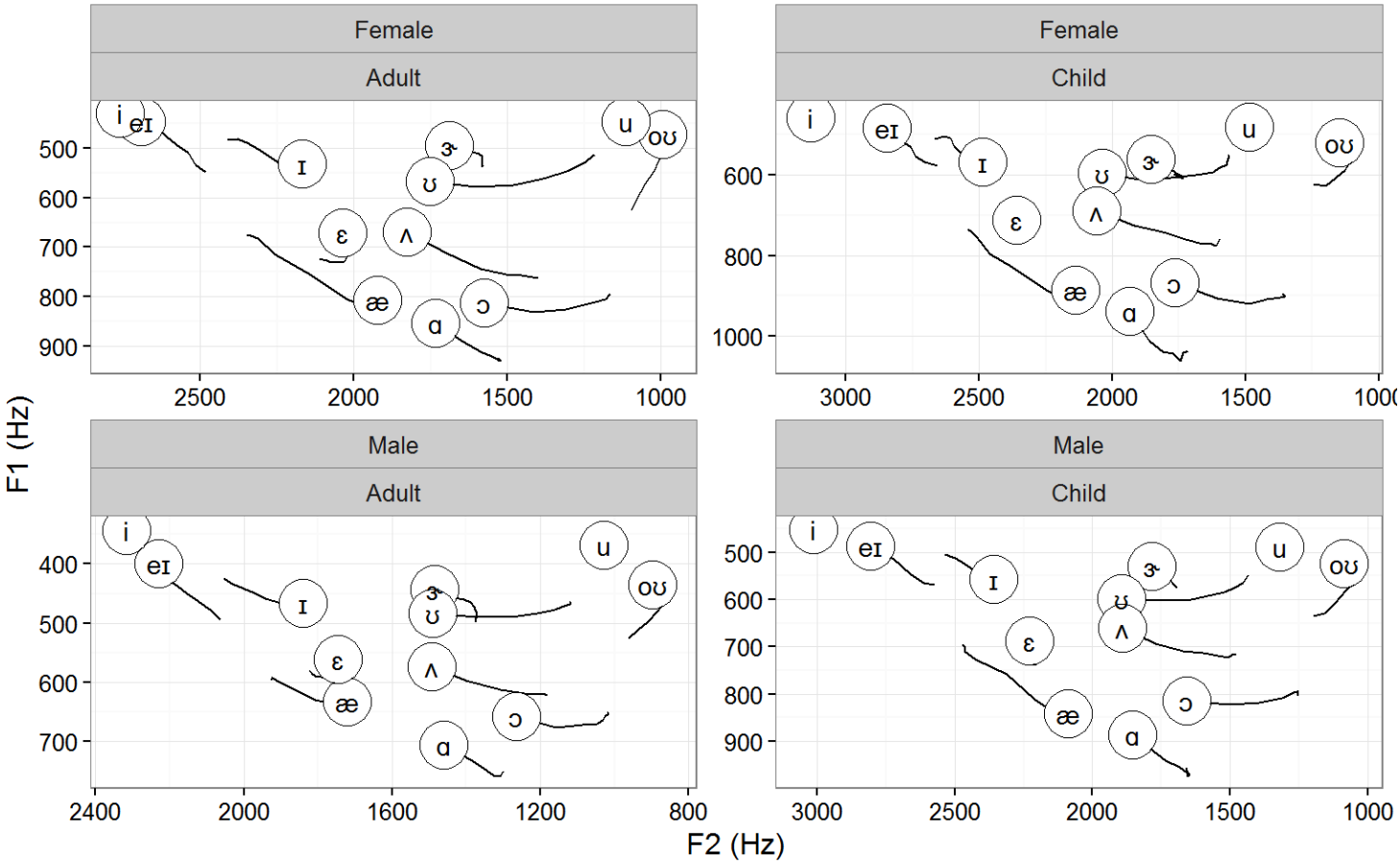
```
px_visc.label <- px_visc +
  geom_label(data=data.wide.visc.sum %>% filter(Time==80),
             aes(label=Vowel.IPA))
px_visc.label
```



Altering the appearance just a bit…

If you don't prefer the style of the new `geom_label`, make the vowel symbols more visible by backdropping a white circle under plain text labels:

```
px_visc.label.circ <- px_visc +
  geom_point(data=data.wide.visc.sum %>% filter(Time==80),
             shape=21, fill="white", size=9)+
  geom_text(data=data.wide.visc.sum %>% filter(Time==80),
             aes(label=Vowel.IPA))
px_visc.label.circ
```



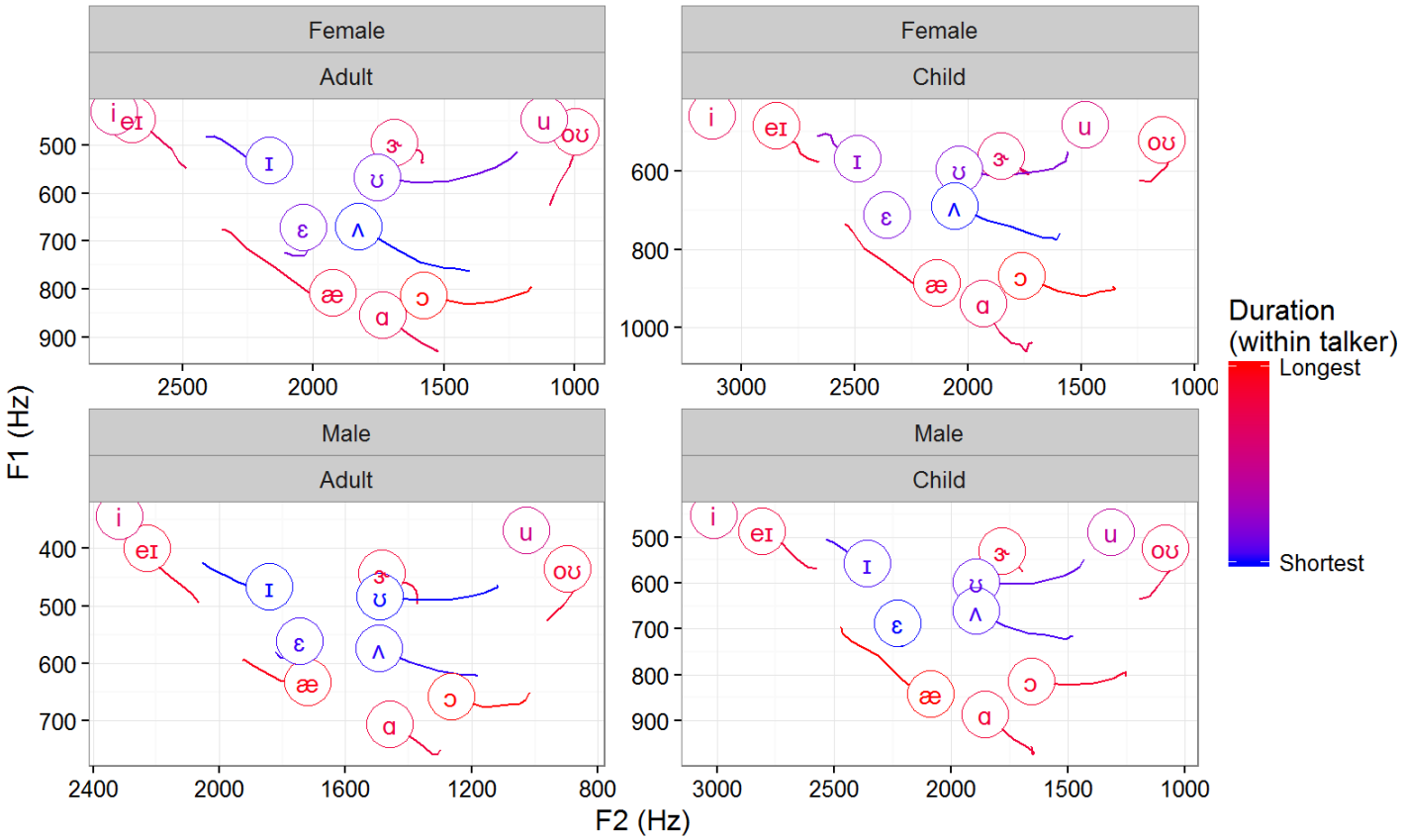
Color each symbol & path

according to normalized vowel duration

Background info: in the first tutorial, we calculated normalized vowel duration as a value between 0 and 1, which represent the shortest and longest measured vowel duration for each talker in the database.

Note how all we're doing is taking the same plot that was just produced, and simply adding a color aesthetic. Since Duration is already in the data frame, it will be *mapped* to the color. We are also creating a custom color scheme that you can change to your liking.

```
px_visc.lab.IPA.dur_color <- px_visc.label.circ+
  aes(color=Duration_norm)+
  scale_color_gradient(name="Duration\n(within talker)",
                       high = "red", low = "blue",
                       breaks=c(0,1),
                       labels=c("Shortest","Longest"))+
  guides(color=guide_colorbar(),
         text="none")
px_visc.lab.IPA.dur_color
```



Now it becomes clear that the vowels that **centralize* are the vowels with the shortest duration. These are the prototypical *lax* vowels in English.

Plot vowel trajectories in an animated sequence:

First, establish the group you want to plot

(because four animated plot panels is too busy); in this case we will plot vowels spoken by women:

```
data_to_plot <- data.wide.visc.sum %>%
  dplyr::filter(Gender=="Female", Age=="Adult")
```

Establish the name of the output file:

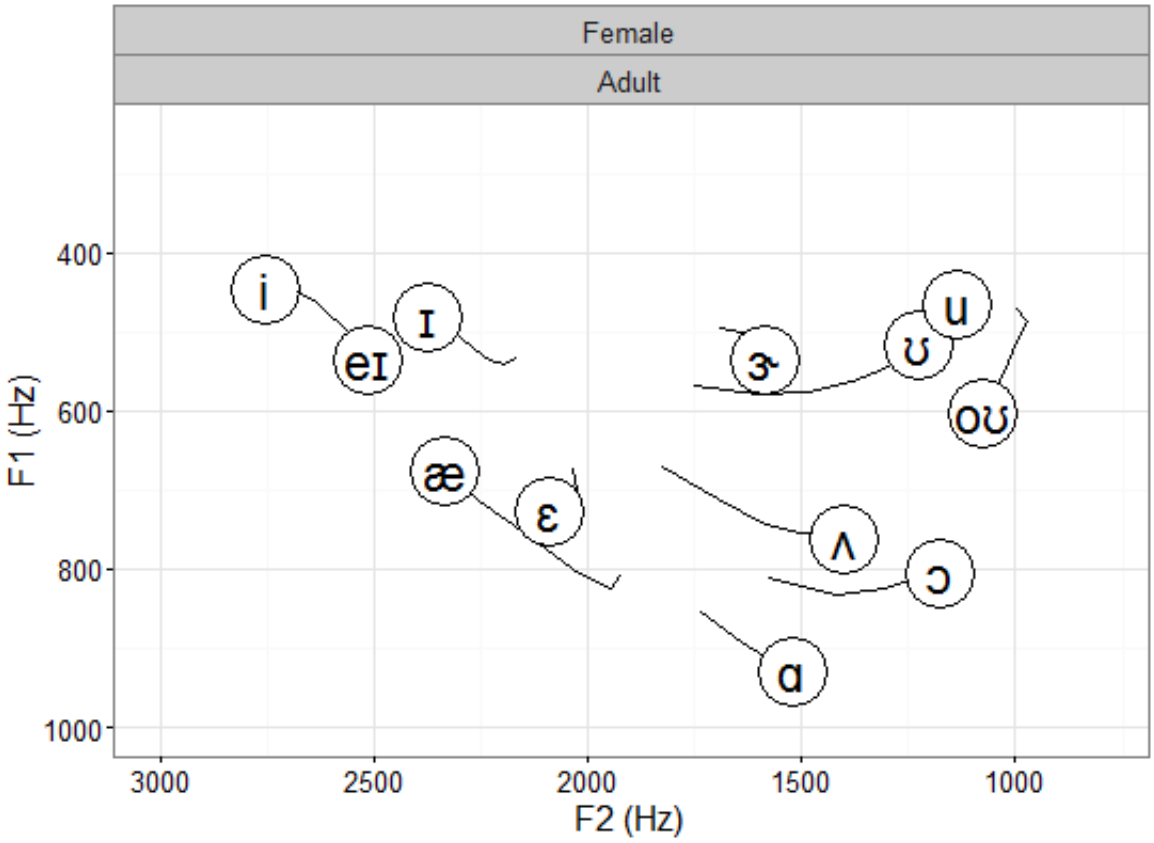
```
movie_filename <- "Vowel_chart_Women.gif"
```

Set some features based on where the formants should be:

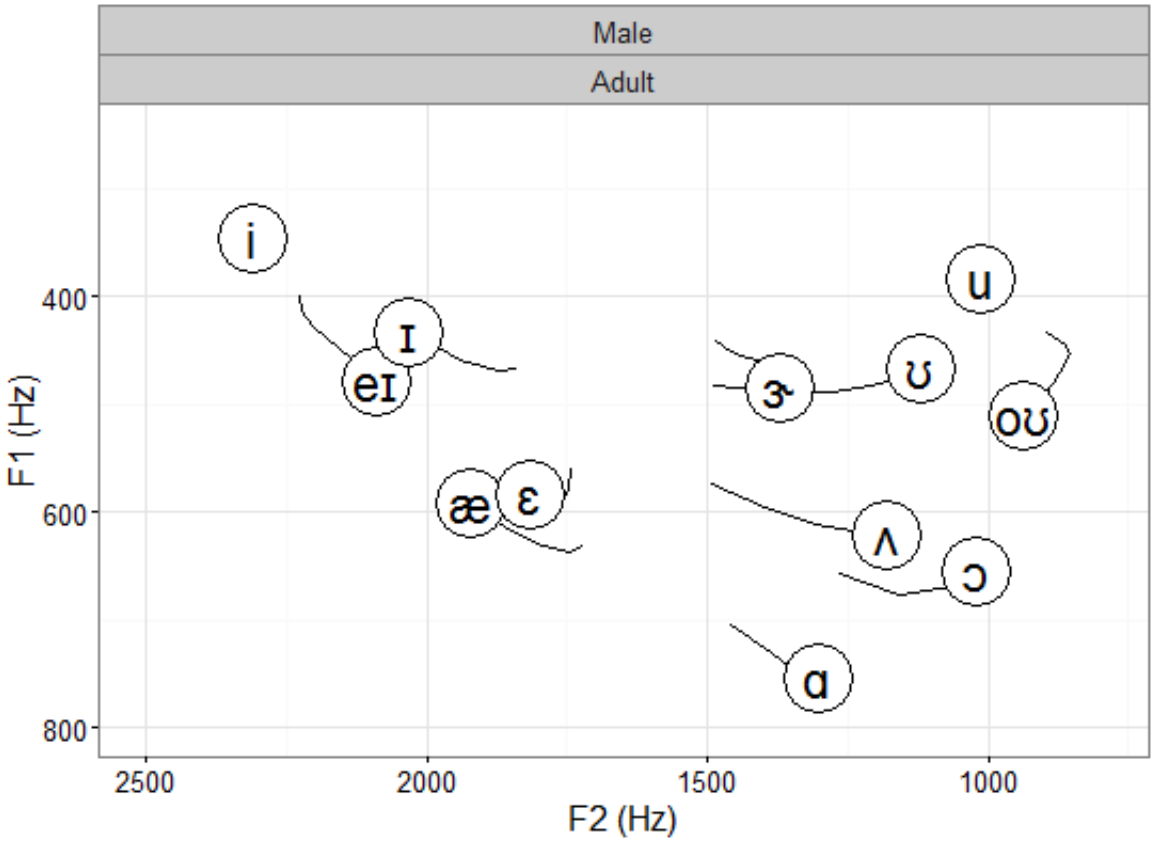
```
fixed_formant_axes <- coord_cartesian(xlim = c(800, 3000), ylim = c(250,1000))
axis_labels <- labs(x="F2 (Hz)", y="F1 (Hz)")

start_time <- 10
end_time <- 80
symbol_size <- 8
```

Here’s the command to make the animation:

```
saveGIF({
  # loop through the vowel chart in time (8 plots)
  for (temp_time in unique(data_to_plot$Time)){
    px_temp <- data_to_plot %>%
      ggplot(.)+
      aes(x=f2, y=f1, group=Vowel)+
      scale_x_reverse()+scale_y_reverse()+
      fixed_formant_axes + axis_labels+
      geom_path()+
      geom_point(data=data_to_plot %>% filter(Time==temp_time),
                 shape=21, fill="white", size=12)+
      geom_text(data=data_to_plot %>% filter(Time==temp_time),
                aes(label=Vowel.IPA))+
      theme_bw()+
      facet_wrap( ~ Gender + Age, scales="free")
    print(px_temp)
  }

  # hold offglide in place, fade out (5 plots)
  for (temp_alpha in seq(1,0,length.out = 4)){
    #     px_fadeout <- data_to_plot[data_to_plot$Time ==end_time,] %>%
    px_fadeout <- data_to_plot %>%
      ggplot(.,
             aes(x=f2, y=f1, group=Vowel))+
      scale_x_reverse()+scale_y_reverse()+
      fixed_formant_axes + axis_labels+
      geom_path(alpha=1)+
      geom_point(data=data_to_plot %>% filter(Time==end_time),
                 alpha=temp_alpha,
                 shape=21, fill="white", size=12)+
      geom_text(data=data_to_plot %>% filter(Time==end_time),
                alpha=temp_alpha,
                aes(label=Vowel.IPA))+
      theme_bw()+
      facet_wrap( ~ Gender + Age, scales="free")
    print(px_fadeout)
  }

},
# intervals: hold first and 8th longer, fly through 9-12
interval = c(0.5, rep(0.2, 6),0.5, rep(0.1,5)),
# interval = 0.2,
ani.height = 400, ani.width = 550,
movie.name = movie_filename
)
```



And here's the same kind of plto for the measurements made in vowel spoken by men:



For simpler (although with less explicit step-wise) animation code, you'll want to check out the "gganimate" extension, available here (https://github.com/dgrtwo/gganimate), and installable using the following code:

```
devtools::install_github("dgrtwo/gganimate")
```

I hope you have enjoyed this tutorial on plotting vowels!

please contact me for any feedback: [MyFirstInitial MyLastName] [at] umn.edu

Matt Winn