Vowel normalization and plotting with the phonR package*

Daniel R. McCloy

Department of Linguistics, University of Washington
drmccloy@uw.edu

1 Introduction

The phonR package is a collection of functions for the R language (R Development Core Team, 2012) intended to meet the needs of phoneticians and phonologists. Currently the package includes functions for vowel normalization and vowel plotting; other functions for tasks like generating standard consonant charts and manipulating distinctive feature vectors for phone inventories are under development. This document describes phonR's vowel plotting and normalization capabilities, and is current as of phonR version 0.3-1.

The vowel plotting capabilities in phonR were developed to enhance the flexibility and visual appeal of vowel plots, especially plots with multiple groups' data on a single graph. To that end, several options are available in phonR that are unavailable or difficult to achieve with other vowel plotting tools (the vowels package in R (Kendall & Thomas, 2009) and its associated website NORM (Thomas & Kendall, 2007) being a popular alternative). In particular, the phonR package implements confidence interval plotting based on bivariate normal density contours, allowing for ellipsoids in F2×F1 space whose major and minor axes are not restricted to the horizontal or vertical (compare to NORM's crosshairs-style plotting of standard deviations in the F1 and F2 dimensions). Additionally, phonR introduces support for automatic drawing of the vowel polygon as well as the option to leave certain vowels unconnected to the polygonal line (e.g., central vowels like [ə]). Finally, both vowel means and individual vowel tokens may be plotted with IPA glyphs or geometric symbols (or may be omitted), with smart defaults for various color, linestyle, and shape options.

The phonR package can be downloaded from the Comprehensive R Archive Network (http://cran.r-project.org/package=phonR). The source code is hosted at https://github.com/drammock/phonR, and is released under GPL-3 (http://www.gnu.org/licenses/gpl.html). The package's ellipse-drawing functionality depends on the R package "mixtools" (Benaglia et al., 2009). Plotting with non-ASCII

*Development of the phonR package was funded in part by the National Institutes of Health, grant # R01DC006014 to Pamela Souza. The author would like to thank August McGrath for her assistance in developing and debugging early versions of the `plotVowels` function.

IPA glyphs instead of geometric shapes requires a typeface with Unicode-IPA symbols; free typefaces with excellent IPA coverage include Charis-SIL (SIL International, 2011), Linux Libertine (Libertine Open Fonts Projekt, 2012), and M+ (Morishita, 2012).

2 Normalizing vowels with phonR

The phonR function `normalizeVowels` implements several of the most common normalization algorithms, which are listed in Table 1. Depending on the normalization method chosen, the function can operate on a single vector of values, or may require both F1 and F2 (Watt-Fabricius method) or all of f0, F1, F2, and F3 (Nearey-2 method). Additionally, the Watt-Fabricius, Nearey-1, Nearey-2 and Lobanov normalization methods can make use of an optional argument `grouping.factor`, which allows normalization to occur speaker-intrinsically (other normalization methods are automatically speaker-intrinsic by virtue of their formulae).

Normalization method	**Source**	**Formula**
Bark	Traunmüller (1990)	$\frac{26.81 \times F_n}{1960 + F_n} - 0.53$
Equivalent Rectangular Bandwidth (erb)	Glasberg & Moore (1990)	$21.4 \times \log_{10}(1 + F_n \times 0.00437)$
Mel	Stevens & Volkmann (1940)	$2595 \times \log_{10}(1 + \frac{F_n}{700})$
Log	n/a	$\log_{10}(F_n)$
Lobanov (z-transform, z-score)	Lobanov (1971)	$\frac{F_n - \mu(F_n)}{\sigma(F_n)}$
Watt-Fabricius (s-centroid)[1]	Watt & Fabricius (2002)	$\frac{F_n}{centroid}$
Nearey-1 (logmean)	Nearey (1977)	$\log_e(F_n) - \mu(\log_e(F_n))$
Nearey-2	Nearey (1977)	$\log_e(F_n) - \sum_{n=0}^{3} \mu(\log_e(F_n))$

Table 1: Normalization methods available via phonR's `normalizeVowels` function

[1] In the phonR package, the centroid is defined as the point $\langle \frac{\min(\overline{F2}_{\text{vowel}}) + \max(\overline{F2}_{\text{vowel}})}{2}, \frac{2 \times \min(\overline{F1}_{\text{vowel}}) + \max(\overline{F1}_{\text{vowel}})}{3} \rangle$. This varies slightly from the formula in Watt & Fabricius (2002), since the phonR implementation simply calculates which vowel has the highest mean F1 value and designates it as low corner of the triangle, rather than asking the user to expressly specify the *trap* or *start* vowel. Similarly, the phonR implementation simply calculates which vowel has the

3 Generating vowel plots with phonR

A comprehensive function call for vowel plotting is shown in Listing 1, and the resulting plot is shown in Figure 1. The basic data are specified with the arguments `f1`, `f2`, and `vowel`, with optional arguments `f3` and `f0` for normalization methods that require them. In Listing 1, the basic data are specified as column names of the data frame given by the `data` argument. Of special note here are the arguments `poly.order` and `poly.include`. The `poly.order` argument specifies the order in which vowels should be connected if `polygon=TRUE`; the default value for this argument is `NULL`, but if left unspecified, the function will attempt to assign a sensible order based on the set {i ɪ e ɛ æ a ɑ ɒ ɔ o ʊ u ʌ ə}. The `poly.include` argument takes an integer indicating how many of the vowels present in the data should be connected by the polygon segments, allowing certain vowels to be omitted (for example, one might choose to connect only the short vowel in a series of long/short vowel pairs, in which case the long vowels should be placed at the end of the `poly.order` vector, and `poly.include` should be the index of the last short vowel).

```
plotVowels(data=myDataFrame, vowel='Vowel_column_name', f1=
  'F1_column_name', f2='F2_column_name', f3='F3_column_name',
  f0='f0_column_name', grouping.factor='Gender_column_name',
  norm.method='mel', match.unit=TRUE, match.axes='absolute',
  points='shape', means='text', points.alpha=0.3, means.alpha=1,
  points.cex=0.6, means.cex=1.2, ignore.hidden=TRUE, ellipses=TRUE,
  ellipse.alpha=0.3173, polygon=TRUE, poly.order=c('i','e','a','o','u'),
  poly.include=NULL, single.plot=TRUE, titles='auto', axis.titles=
  'auto', axis.cex=0.8, garnish.col='#666666FF', grayscale=FALSE,
  vary.colors=TRUE, vary.shapes=TRUE, vary.lines=TRUE, legend='TRUE',
  output='pdf', family='FreeSerif', pointsize=12, units='in',
  width=6.5, height=6.5, res=72, asp=NULL)
```

Listing 1: A comprehensive function call to phonR's `plotVowels` function

The argument `garnish.col` controls the color of axis lines, tickmarks, tickmark labels, and annotations. It defaults to a gray color to help ensure that the visual emphasis of the plot is the data itself. Also shown in Figure 1 is the automatic axis labeling with regard to the chosen normalization method, variation in plot symbol shape and linestyle, and the effect of the transparency arguments (`points.alpha` and `means.alpha`).[2] The line styles cycle through a custom set of eleven, but the plot symbols are restricted to a subset of the symbols available through `pch`, namely the set ● ○ ▲ △ ■ □ ◆ ◇ + × ▽.

highest mean F2 value and uses that to calculate the upper left corner, rather than expressly looking for the mean of the "point-vowel" /i/. The upper right corner is, as in the original method, derived from the other two.

[2]Note, however, that the argument `ellipse.alpha` is not a transparency level, but an α-level in the statistical sense. Its default value of 0.3173 corresponds to an ellipse encompassing 68.27% of the data points, equivalent to ±1 standard deviation from the bivariate mean. Conventional 95% confidence ellipses are easily achieved by setting `ellipse.alpha=0.05`.

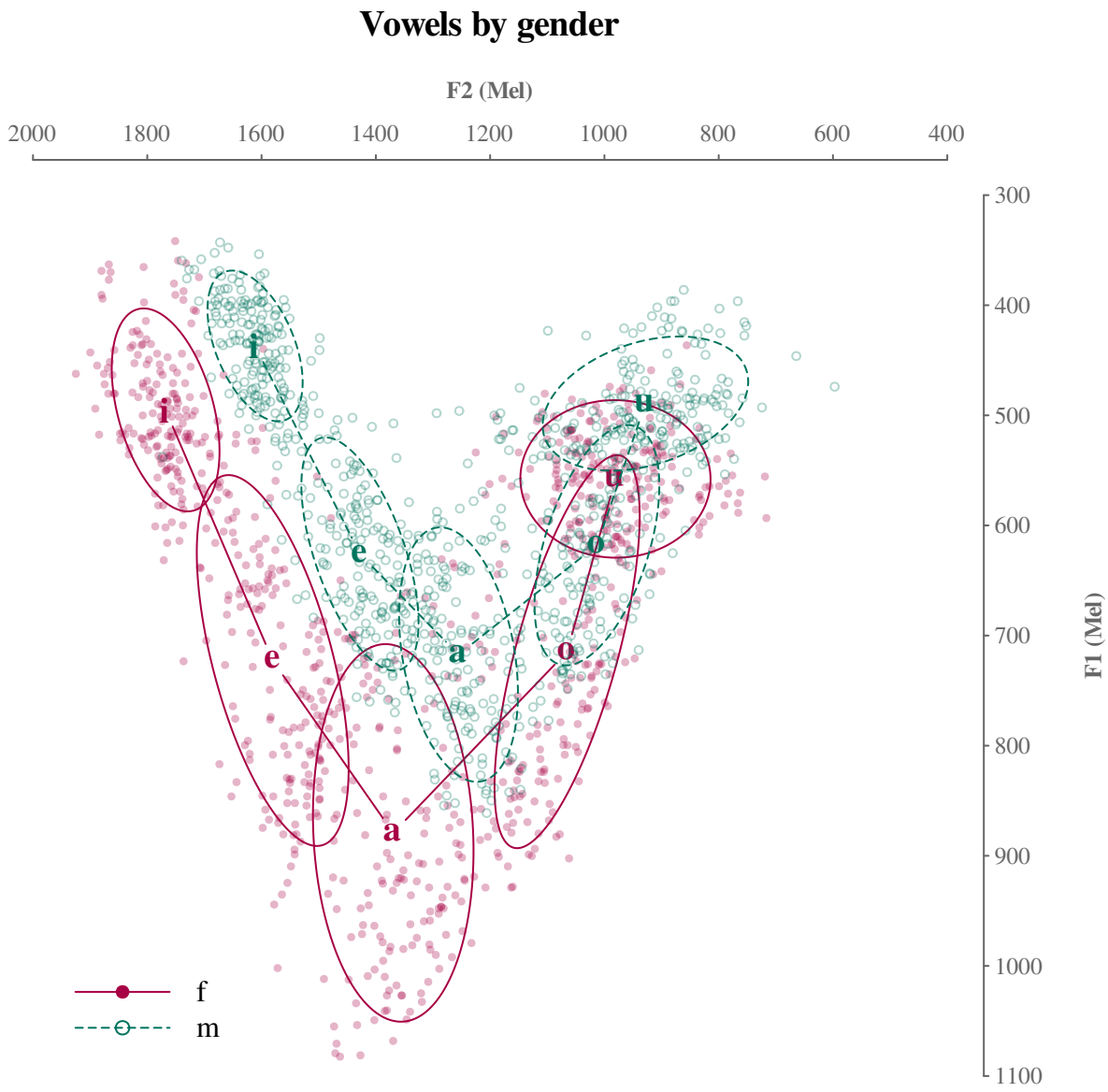

Figure 1: Vowel plot based on the maximal function call in Listing 1

A more typical function call includes fewer arguments, as in Listing 2; the output of this function call is shown in Figure 2. Notable here is the fact that the `f1`, `f2`, and `vowel` arguments may be omitted if an argument for `data` is supplied that contains columns labeled "f1", "f2", and "vowel", respectively. Listing 2 also illustrates a few of the intelligent defaults of phonR's `plotVowels` function. For example, graphical options like line style or shape are held constant by default if the plot is in color. Colored plots use colors of equal chroma and luminance, with hues equally spaced around the color circle in HCL space (Zeileis et al., 2009).

```
plotVowels(data=indo, grouping.factor='subj', norm.method='bark',
  match.unit=FALSE, points='none', means='text', ignore.hidden=TRUE,
  ellipses=FALSE, polygon=TRUE, single.plot=TRUE, grayscale=FALSE,
  titles='Indonesian vowel spaces (means and polygons)', asp=1.5,
  legend='bottomright', family='FreeSerif', output='pdf')
```

Listing 2: A typical function call to phonR's `plotVowels` function

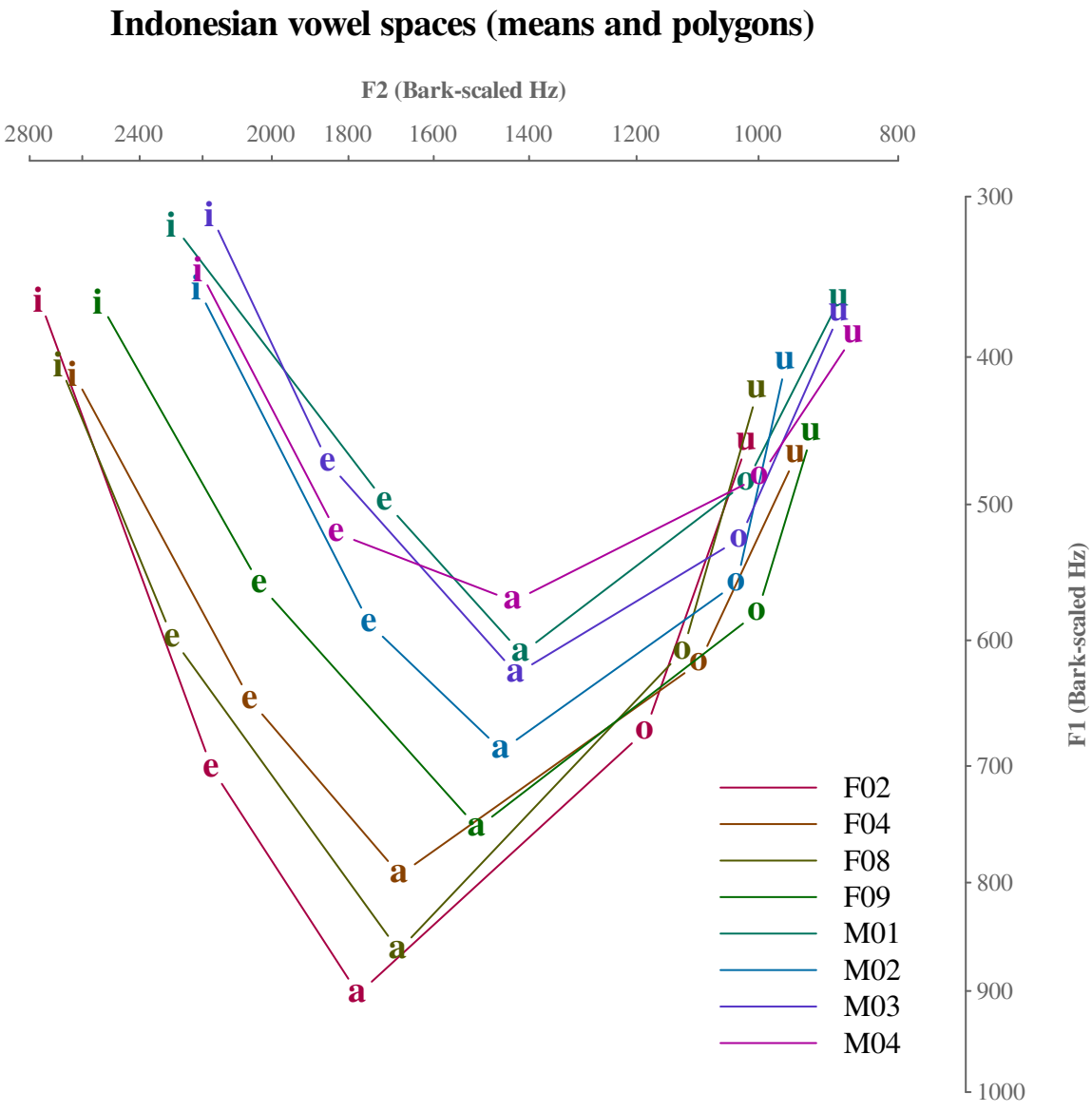

Figure 2: Vowel plot based on the typical function call in Listing 2

Custom titles can also be included; the `titles` argument accepts a vector of strings and will recycle or discard strings (with a warning to the user) if there is a mismatch between number of graphs and number of strings provided. Figure 2 also illustrates the `match.unit=FALSE` argument, which transforms the data points using the specified normalization method (in this case, the Bark transformation) and plots them in the transformed space, but labels the axes with values equispaced in Hz, highlighting the non-linearity of the transform (note the unequal spacing of tickmarks that are numerically equidistant). Finally, note that the `legend` argument takes either boolean values or the special location arguments of R's legend() function (e.g., 'topleft').

A minimal function call may include only `f1`, `f2`, and `vowel` arguments. Note that in Listing 3 the arguments `f1`, `f2`, and `vowel` are supplied without quotation marks when supplied as vectors (rather than as column names of the `data` argument). Omitting `grouping.factor` treats all points as belonging to the same subject or group. The four main plot elements (points, means, ellipses, and polygons) each have default values and thus may be omitted if the defaults are suitable. Figure 3 (corresponding to Listing 3) shows the effect of the `match.axes` argument, which determines whether all graphs will have the same axis limits; possible values are `'absolute'` (all graphs have the same dimensions), `'relative'` (all graphs have the same scale and aspect ratio, but their absolute limits may vary), or `'none'` (graphs may vary in their scale, aspect ratio, and/or absolute limits). This argument applies only when a grouping factor is supplied and `single.plot=FALSE`, so it is ignored (coerced to `'absolute'`) when `single.plot=TRUE`.

```
plotVowels(f1=f1Vector, f2=f2Vector, vowel=vowelVector,
  grouping.factor=groupingVector, match.axes=TRUE,
  points='none', means='text', single.plot=FALSE,
  family='FreeSerif', output='screen')
```

Listing 3: A minimal function call to phonR's `plotVowels` function

Figure 3 also shows another unique feature of the `plotVowels` function, which is that the argument `single.plot=FALSE` behaves differently depending on output method. If the output method is `'screen'` then the graphs for each group are arranged in a grid so they can be easily inspected side-by-side. In contrast, if the output method is `'pdf'`, `'svg'`, `'tif'`, `'png'`, `'jpg'`, or `'bmp'`, then a separate file will be created for each group. Figures 3 also illustrates the automatic titling functionality, which labels each graph with the grouping factor value (in this case, subject number, though this could just as easily be task type, dialect, ethnicity, etc). Finally, as mentioned above, Figure 3 illustrates the effect of the argument `match.axes='absolute'`, which forces all graphs to the same dimensions and axis limits, allowing visual comparisons between graphs of the relative size and position of different speakers' vowel spaces.

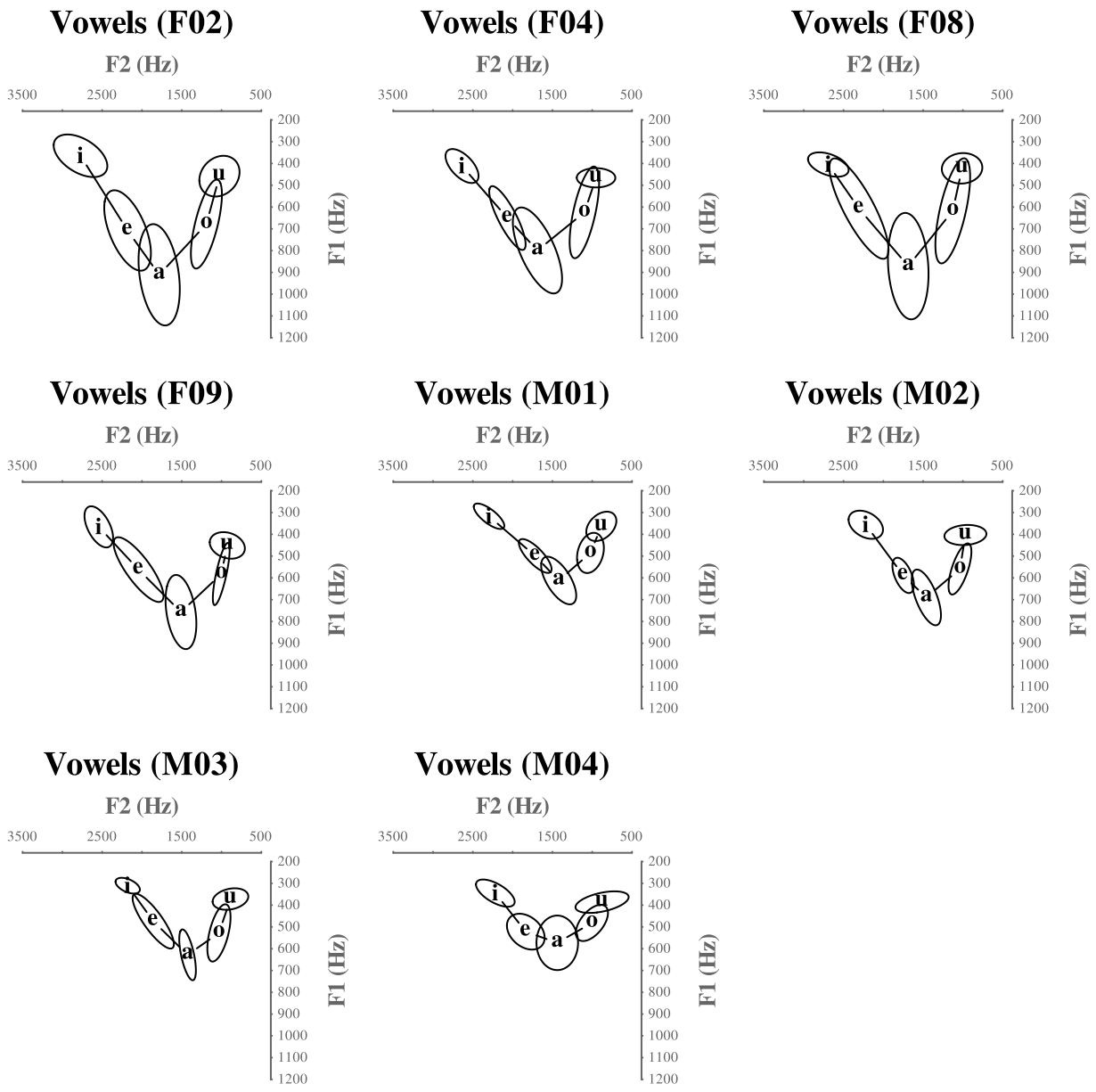

Figure 3: Vowel plot based on the minimal function call in Listing 3

4 Known issues

There are two major (known) shortcomings with the `plotVowels` function. The first is its inability to handle diphthong data; this functionality is the current focus of development. The second issue regards onscreen plotting: the default R onscreen graphics windows typically have a series of menu commands including commands like "File > Save as > PDF". Because of the way that fonts are specified, plotting using `output='screen'` will display fonts correctly in the onscreen

window, but fonts will often fail to be embedded in vector output formats when using the onscreen window's "Save as" menu commands. For this reason, it is recommended that the onscreen display be used to fine-tune the appearance of plots, and the other output choices (`'pdf'`, `'tif'`, etc.) be used when exporting finished plots for use in publications, presentations, or webpages.

References

Benaglia, T., Chauveau, D., Hunter, D. R., & Young, D. (2009). mixtools: An R package for analyzing finite mixture models. *Journal of Statistical Software*, *32*(6), 1–29.

Glasberg, B. R., & Moore, B. C. J. (1990). Derivation of auditory filter shapes from notched-noise data. *Hearing Research*, *47*(1-2), 103–138.

Kendall, T., & Thomas, E. R. (2009). Vowels: Vowel manipulation, normalization, and plotting in R. http://cran.r-project.org/web/packages/vowels/index.html.

Libertine Open Fonts Projekt (2012). Linux Libertine. http://www.linuxlibertine.org/.

Lobanov, B. M. (1971). Classification of Russian vowels spoken by different speakers. *The Journal of the Acoustical Society of America*, *49*(2), 606–608.

Morishita, C. (2012). M+ Fonts. http://mplus-fonts.sourceforge.jp/.

Nearey, T. M. (1977). *Phonetic feature systems for vowels*. Doctoral dissertation, University of Alberta.

R Development Core Team (2012). R: A language and environment for statistical computing. http://www.R-project.org/.

SIL International (2011). Charis SIL. http://scripts.sil.org/CharisSILfont.

Stevens, S. S., & Volkmann, J. (1940). The relation of pitch to frequency: A revised scale. *The American Journal of Psychology*, *53*(3), pp. 329–353.

Thomas, E. R., & Kendall, T. (2007). NORM: The vowel normalization and plotting suite. http://ncslaap.lib.ncsu.edu/tools/norm/index.php.

Traunmüller, H. (1990). Analytical expressions for the tonotopic sensory scale. *The Journal of the Acoustical Society of America*, *88*(1), 97–100.

Watt, D., & Fabricius, A. H. (2002). Evaluation of a technique for improving the mapping of multiple speakers' vowel spaces in the F1 ~ F2 plane. *Leeds Working Papers in Linguistics and Phonetics*, *9*, 159–173.

Zeileis, A., Hornik, K., & Murrell, P. (2009). Escaping RGBland: Selecting colors for statistical graphics. *Computational Statistics & Data Analysis*, *53*(9), 3259–3270.