# PlotPolarGAMM_demo

### Benson

### 3/18/2024

`PlotPolarGAMM` is a function that allows you to visualize and plot `Plotly` figures of GAMM-fitted ultrasound tongue contours with provided GAMM models.

```
plotPolarGAMM(
model,
targetName,
targetL,
rmax,
view='X',
showLegend=T,
showDiff=T,
fontSize=25,
returnData=F
)
```

## Arguments

### model

A GAMM model, resulting from the functions gam or bam.

### target

The name of the categories of different types of sounds (e.g. *vowel* for tokens of /i/, /a/, and /u/, or *place* for alveolar and velar sounds.)

### targetL

The categories of the target (e.g. use c('i', 'a', 'u') for the *vowel* example previously, or c('velar', 'alvolar') for the *place* example.)

### rmax

The maximum radius of the plot.

### view

The predictor, usually 'X'.

### showLegend

Whether to show the legend.

### showDiff

Whether to plot the different areas, indicated by gray color blocks.

### fontSize

The font size.

### returnData

Whether to return the fitted data as a dataframe. If False, it will return a Plotly plot.

# Import `plot_polar_GAMM` and the other two modules used to perform polar GAMM.

```
source('plot_polar_GAMM.R')
library(itsadug)
```

```
## Loading required package: mgcv
```

```
## Loading required package: nlme
```

```
## This is mgcv 1.8-38. For overview type 'help("mgcv-package")'.
```

```
## Loading required package: plotfunctions
```

```
## Loaded package itsadug 2.4 (see 'help("itsadug")' ).
```

```
library(rticulate)
```

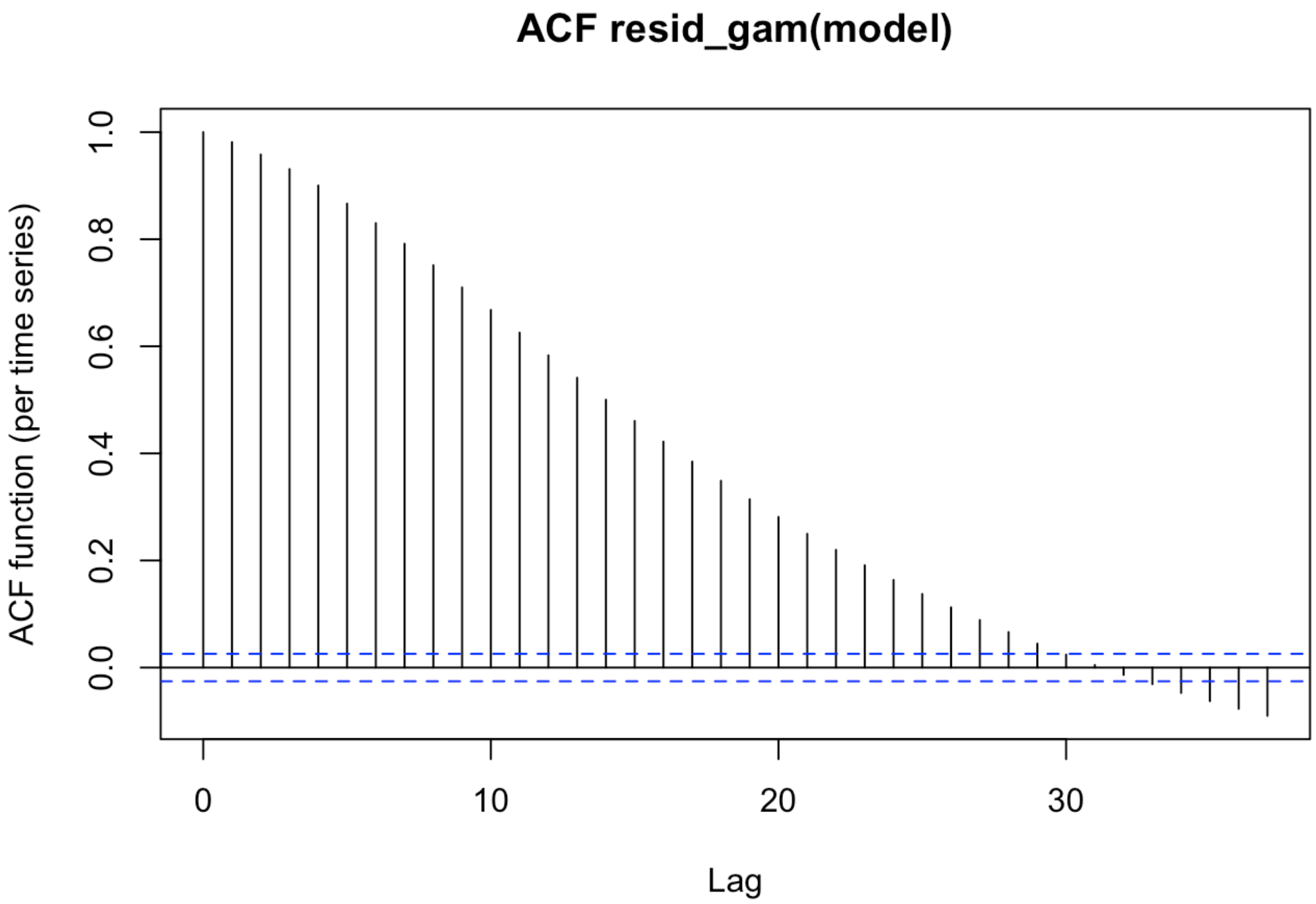
# Import the ultrasound contour data. In `contours_demo.csv`, there are the tongue contour tracings of /i/, /u/, and /y/ at the mid time point.

```
df <- read.csv('contours_demo.csv')
df$vowel <- factor(df$vowel)
df$token <- factor(df$token)
df$X <- df$x
df$Y <- df$y
df <- start_event(df, event=c('token', 'vowel'), column="time_point")
```

# Construct teh GAMM model.

```
model <- polar_gam(
  Y~ vowel +
    s(X, by=vowel) +
    s(token, bs='re'),
  data=df,
  origin=c(562, 573)
)

model.Acf <- acf_resid(model)
```



```
model <- polar_gam(
  Y ~ vowel +
    s(X, by=vowel) +
    s(token, bs='re'),
  data=df,
  origin=c(562, 573),
  rho=model.Acf[2],
  AR.start=df$start.event
)
```

## Put the model into the function, and you will get the plot.

```
plotPolarGAM(model=model,
             targetName='vowel',
             targetL=c('i', 'u', 'y'),
             rmax=max(df$Y)*1.05,
             view='X',
             showDiff=F,
             returnData=F)
```

```
## Loading required package: gss
```

```
## Loading required package: ggplot2
```

```
##
## Attaching package: 'ggplot2'
```

```
## The following object is masked from 'package:plotfunctions':
##
##     alpha
```

```
## Loading required package: plotly
```

```
##
## Attaching package: 'plotly'
```

```
## The following object is masked from 'package:ggplot2':
##
##     last_plot
```

```
## The following object is masked from 'package:plotfunctions':
##
##     add_bars
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

```
## The following object is masked from 'package:graphics':
##
##     layout
```

```
## Loading required package: sets
## Registered S3 method overwritten by 'sets':
##   method        from
##   print.element ggplot2
```

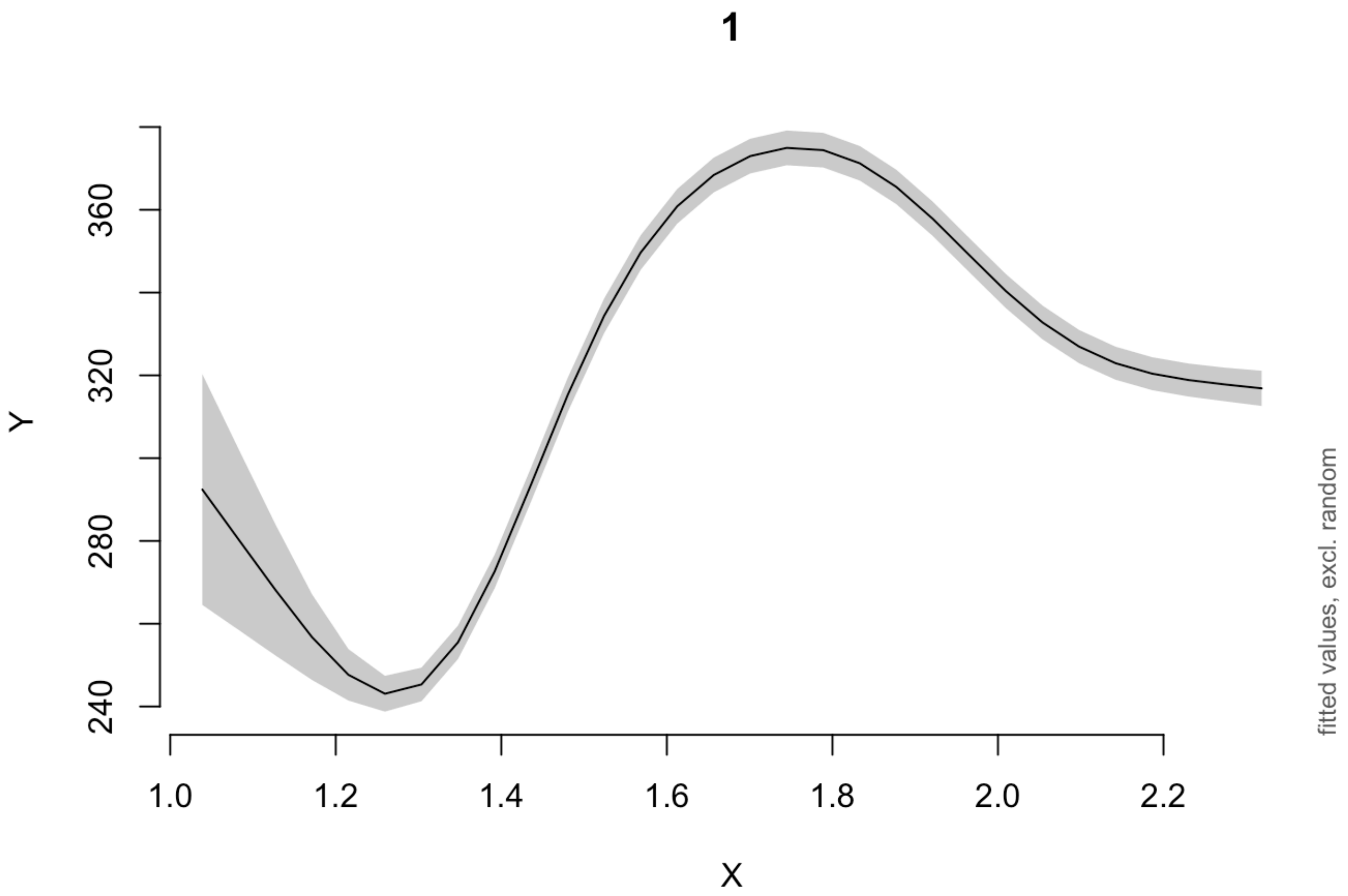
```
##
## Attaching package: 'sets'
```

```
## The following object is masked from 'package:plotly':
##
##     %>%
```

```
## Loading required package: RColorBrewer
```

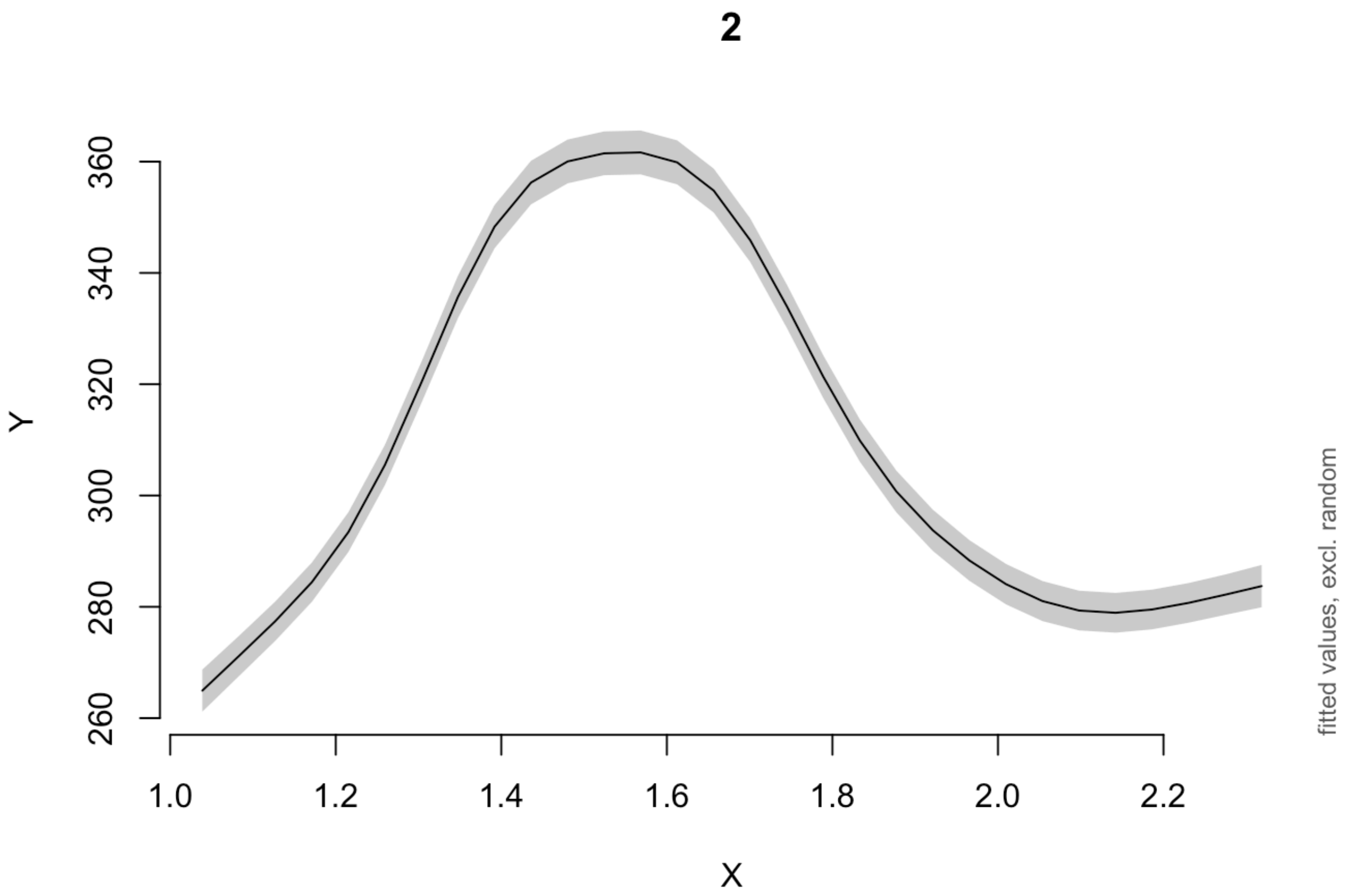
```
## Summary:
##  * vowel : factor; set to the value(s): i.
##  * X : numeric predictor; with 30 values ranging from 1.038784 to 2.318545.
##  * token : factor; set to the value(s): 0. (Might be canceled as random effect,
check below.)
##  * NOTE : The following random effects columns are canceled: s(token)
##
```

```
## Warning in plotL[i] <- plot_smooth(model, view = view, cond =
## changeName(list(c(targetL[i])), : number of items to replace is not a multiple
## of replacement length
```



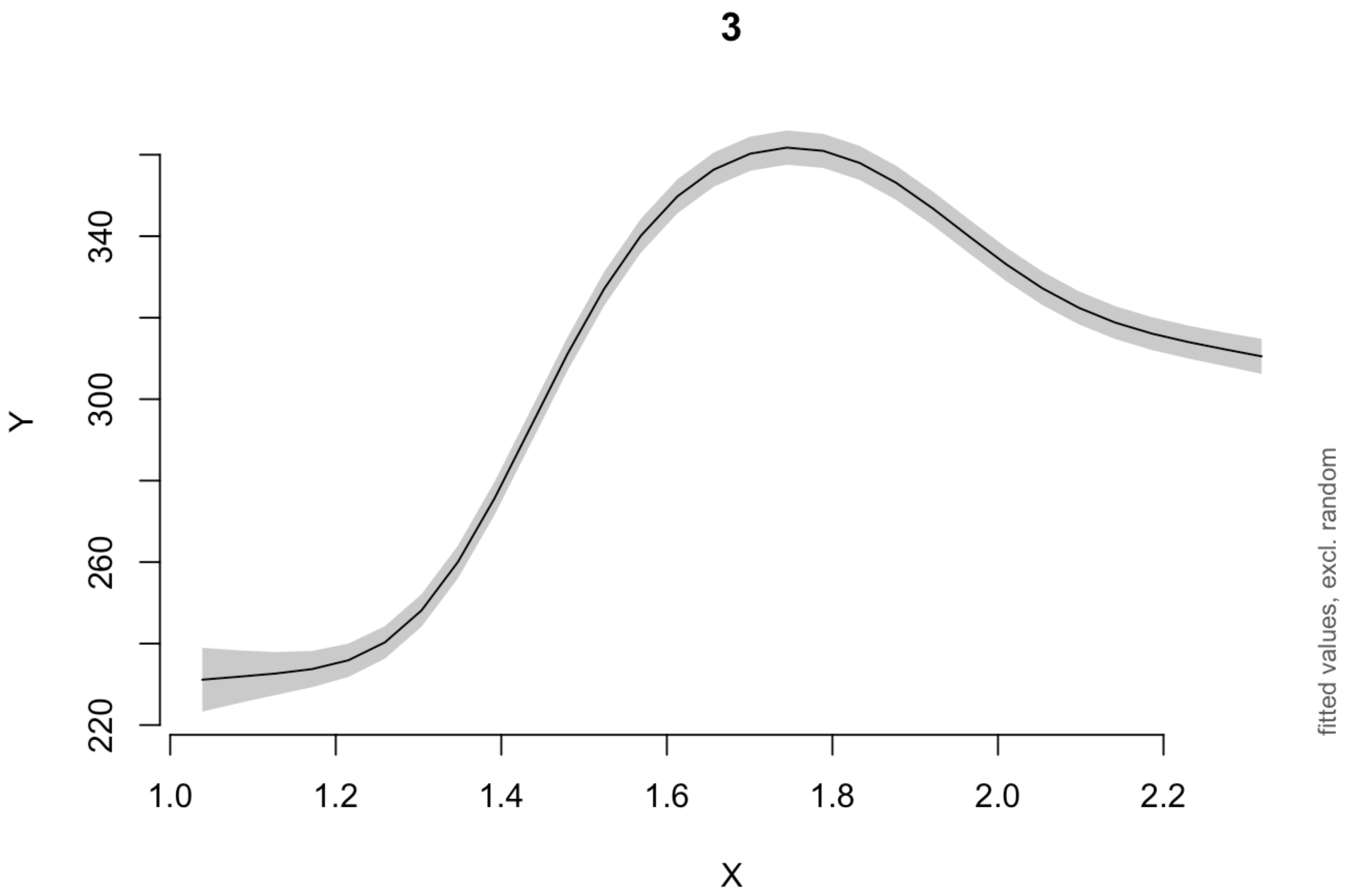
```
## Summary:
##  * vowel : factor; set to the value(s): u.
##  * X : numeric predictor; with 30 values ranging from 1.038784 to 2.318545.
##  * token : factor; set to the value(s): 0. (Might be canceled as random effect,
check below.)
##  * NOTE : The following random effects columns are canceled: s(token)
##
```

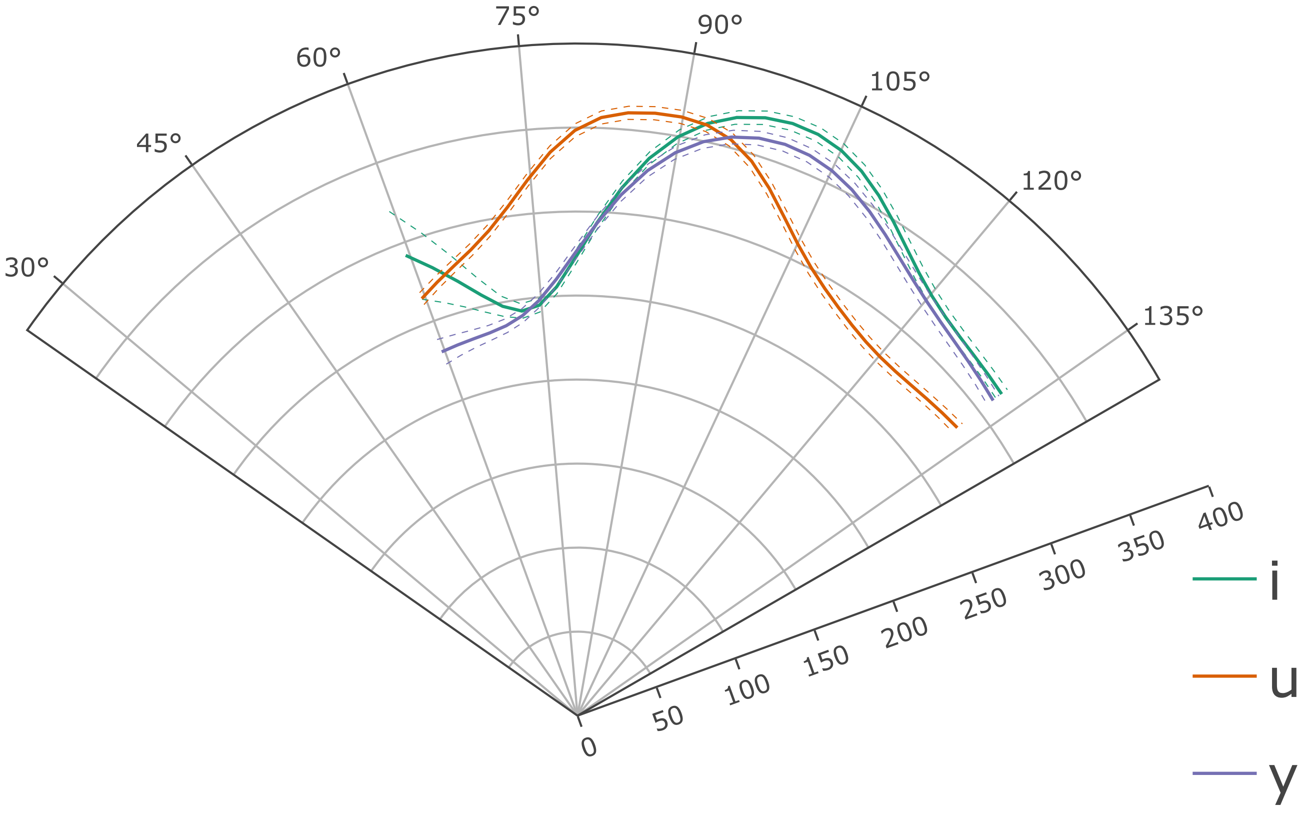
```
## Warning in plotL[i] <- plot_smooth(model, view = view, cond =
## changeName(list(c(targetL[i])), : number of items to replace is not a multiple
## of replacement length
```



```
## Summary:
##  * vowel : factor; set to the value(s): y.
##  * X : numeric predictor; with 30 values ranging from 1.038784 to 2.318545.
##  * token : factor; set to the value(s): 0. (Might be canceled as random effect,
check below.)
##  * NOTE : The following random effects columns are canceled: s(token)
##
```

```
## Warning in plotL[i] <- plot_smooth(model, view = view, cond =
## changeName(list(c(targetL[i])), : number of items to replace is not a multiple
## of replacement length
```


3
Y
340
300
260
220
1.0
1.2
1.4
1.6
1.8
2.0
2.2
X
fitted values, excl. random

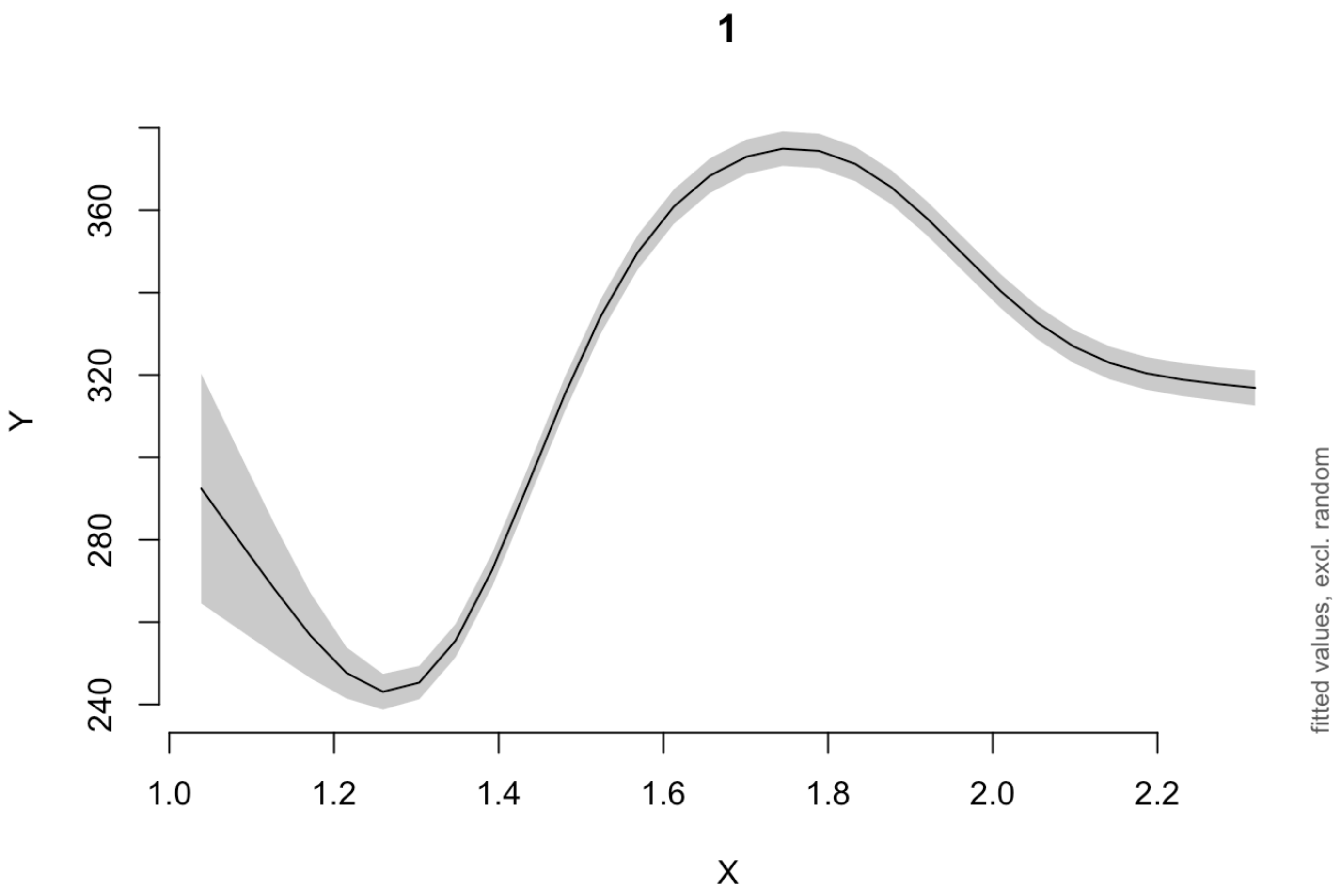

30°
45°
60°
75°
90°
105°
120°
135°
0
50
100
150
200
250
300
350
400
i
u
y

# You can also set `returnData` to `True`, and get the estimated contours.

```
df <- plotPolarGAM(model=model,
                   targetName='vowel',
                   targetL=c('i', 'u', 'y'),
                   rmax=max(df$Y)*1.05,
                   view='X',
                   showDiff=F,
                   returnData=T)
```

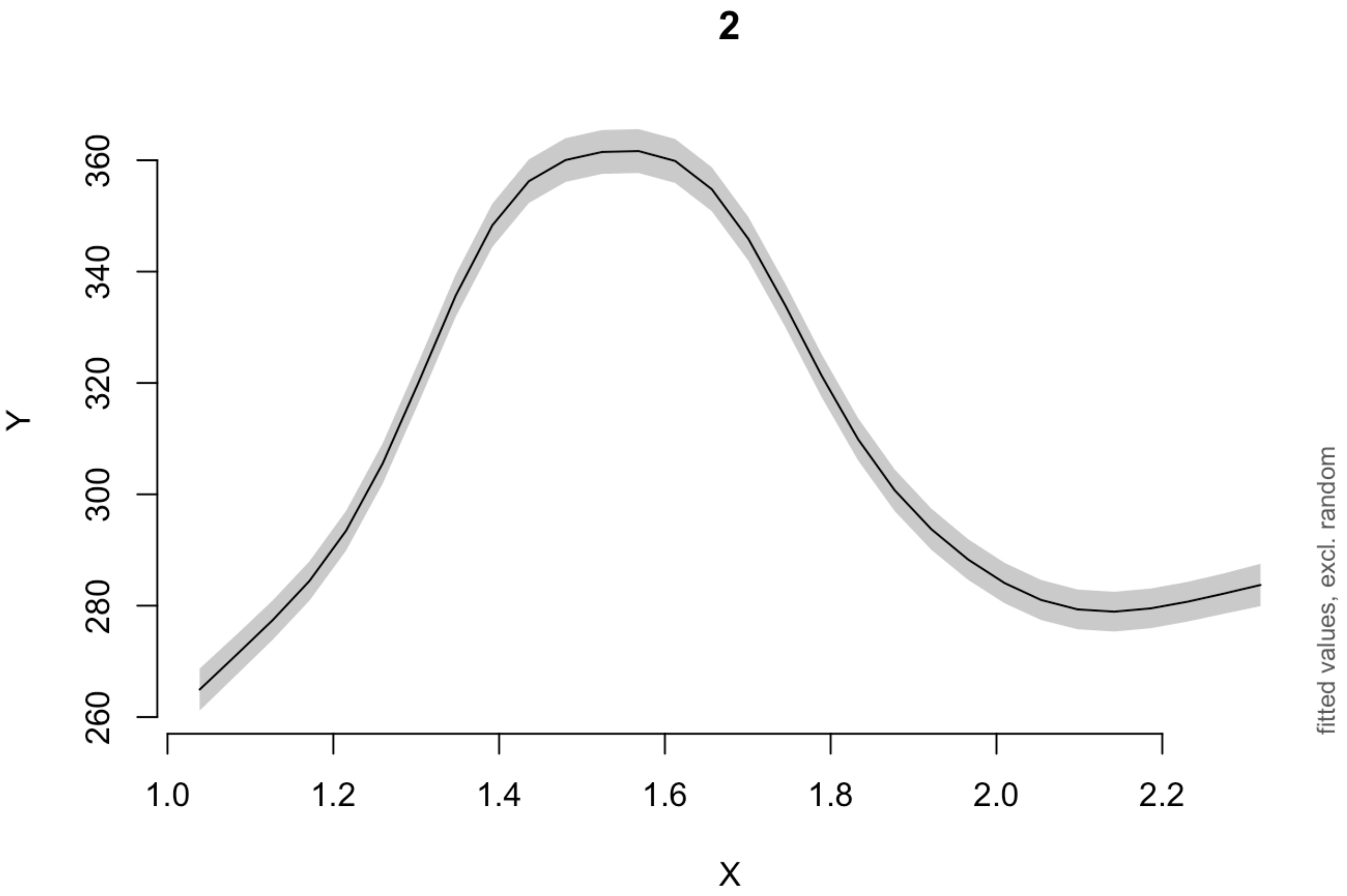
```
## Summary:
##  * vowel : factor; set to the value(s): i.
##  * X : numeric predictor; with 30 values ranging from 1.038784 to 2.318545.
##  * token : factor; set to the value(s): 0. (Might be canceled as random effect,
check below.)
##  * NOTE : The following random effects columns are canceled: s(token)
##
```

```
## Warning in plotL[i] <- plot_smooth(model, view = view, cond =
## changeName(list(c(targetL[i])), : number of items to replace is not a multiple
## of replacement length
```



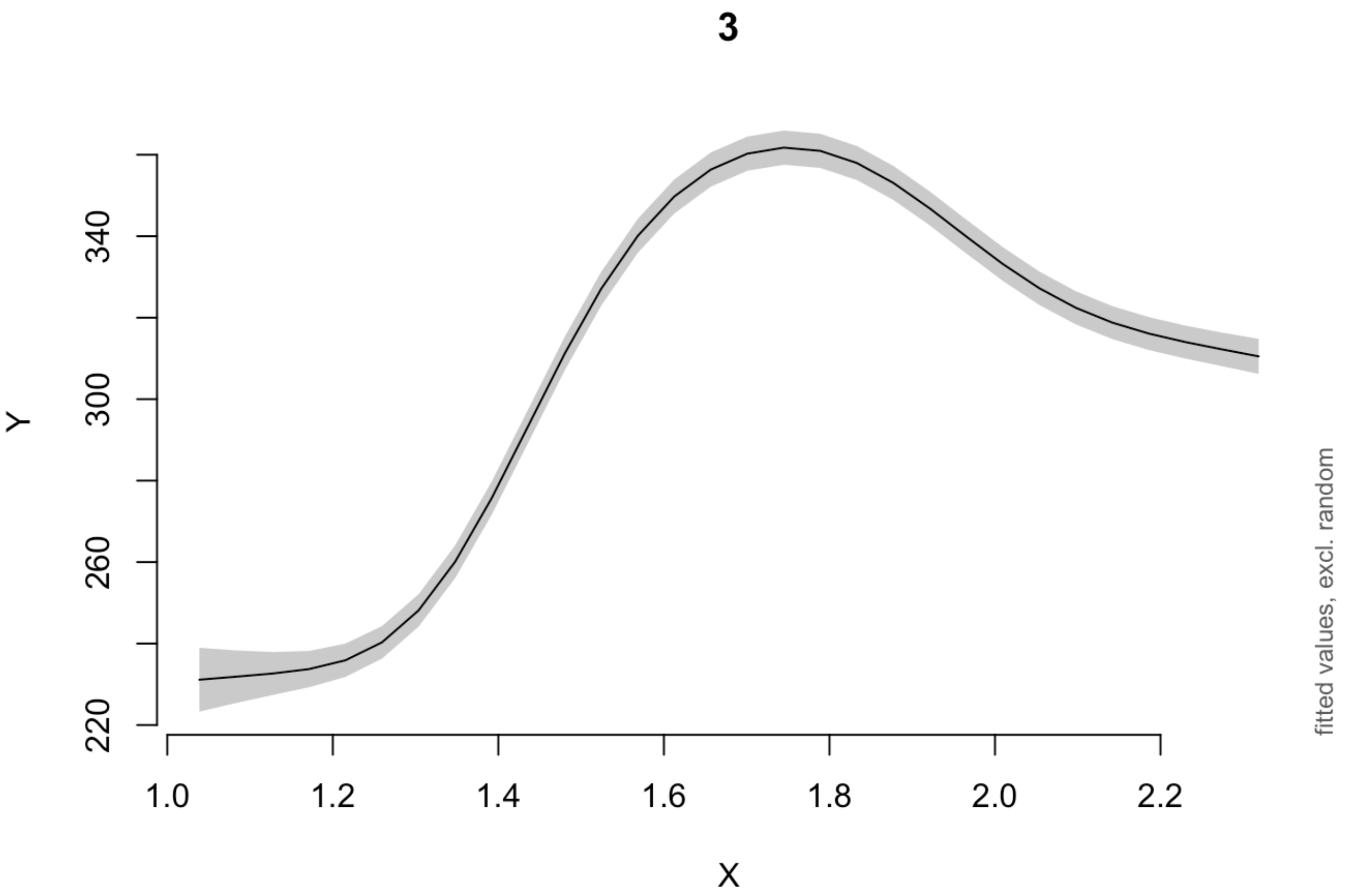
```
## Summary:
##  * vowel : factor; set to the value(s): u.
##  * X : numeric predictor; with 30 values ranging from 1.038784 to 2.318545.
##  * token : factor; set to the value(s): 0. (Might be canceled as random effect,
check below.)
##  * NOTE : The following random effects columns are canceled: s(token)
##
```

```
## Warning in plotL[i] <- plot_smooth(model, view = view, cond =
## changeName(list(c(targetL[i])), : number of items to replace is not a multiple
## of replacement length
```



```
## Summary:
##  * vowel : factor; set to the value(s): y.
##  * X : numeric predictor; with 30 values ranging from 1.038784 to 2.318545.
##  * token : factor; set to the value(s): 0. (Might be canceled as random effect,
check below.)
##  * NOTE : The following random effects columns are canceled: s(token)
##
```

```
## Warning in plotL[i] <- plot_smooth(model, view = view, cond =
## changeName(list(c(targetL[i])), : number of items to replace is not a multiple
## of replacement length
```



```
write.csv(df, 'results.csv')
```