Supplementary Materials for Revealing perceptual structure through input variation: cross-accent categorization of vowels in five accents of English

R Markdown authors: Jason A. Shaw & Jen Hay

3/29/2023

Description

This is an R Markdown document for reproducing analysis and figures appearing in the *Laboratory Phonology* article entitled, "Revealing perceptual structure through input variation: cross-accent categorization of vowels in five accents of English"

Preliminaries: Required packages

```
library(dendextend)
library(MASS)
library(knitr)
library(kableExtra)
```

Define confusion matrix function

We define a function for constructing confusion matrices from our raw data using the table command.

```
conmtrx <- function(resp,corresp) {
  ref = matrix(resp, ncol=1)
  pred = matrix(corresp, ncol=1)
  confusion_matrix <- table(ref, pred)
  confusion_matrix
}
```

Load data

```
data <- read.csv(file.choose()) #filename: LabPhon_data_2023.csv
```

Subset the data by condition. There are 13 conditions in total.

```
# within-accent conditions
AA_A = subset(data, condition == 'AA' & segmentPosition == 'NoLTrialList'& listener_group == 'A')
NN_N = subset(data, condition == 'NN' & segmentPosition == 'NoLTrialList'& listener_group == 'N')
LL_L = subset(data, condition == 'LL' & segmentPosition == 'NoLTrialList'& listener_group == 'L')
YY_Y = subset(data, condition == 'YY' & segmentPosition == 'NoLTrialList'& listener_group == 'Y')
ZZ_Z = subset(data, condition == 'ZZ' & segmentPosition == 'NoLTrialList'& listener_group == 'Z')

# stimulus effect conditions
```

```
NA_N = subset(data, condition == 'NcA' & segmentPosition == 'NoLTrialList'& listener_group == 'N')
LA_L = subset(data, condition == 'LA' & segmentPosition == 'NoLTrialList'& listener_group == 'L')
YA_Y = subset(data, condition == 'YA' & segmentPosition == 'NoLTrialList'& listener_group == 'Y')
ZA_Z = subset(data, condition == 'ZA' & segmentPosition == 'NoLTrialList'& listener_group == 'Z')

# Conditions from Shaw et al 2018 reanalyzed with new methods
AN_A = subset(data, condition == 'AN' & segmentPosition == 'NoLTrialList'& listener_group == 'A')
AL_A = subset(data, condition == 'AL' & segmentPosition == 'NoLTrialList'& listener_group == 'A')
AY_A = subset(data, condition == 'AY' & segmentPosition == 'NoLTrialList'& listener_group == 'A')
AZ_A = subset(data, condition == 'AZ' & segmentPosition == 'NoLTrialList'& listener_group == 'A')
```

Build confusion matrices

We use the function defined above to make a confusion matrix for each condition. As is standard, we have the labels (response categories) in columns and the stimulus items in rows.

```
AA_A.m<-conmtrx(AA_A$stimulusWord, AA_A$selectedWord)
NN_N.m<-conmtrx(NN_N$stimulusWord, NN_N$selectedWord)
LL_L.m<-conmtrx(LL_L$stimulusWord, LL_L$selectedWord)
YY_Y.m<-conmtrx(YY_Y$stimulusWord, YY_Y$selectedWord)
ZZ_Z.m<-conmtrx(ZZ_Z$stimulusWord, ZZ_Z$selectedWord)

NA_N.m<-conmtrx(NA_N$stimulusWord, NA_N$selectedWord)
LA_L.m<-conmtrx(LA_L$stimulusWord, LA_L$selectedWord)
YA_Y.m<-conmtrx(YA_Y$stimulusWord, YA_Y$selectedWord)
ZA_Z.m<-conmtrx(ZA_Z$stimulusWord, ZA_Z$selectedWord)

AN_A.m<-conmtrx(AN_A$stimulusWord, AN_A$selectedWord)
AL_A.m<-conmtrx(AL_A$stimulusWord, AL_A$selectedWord)
AY_A.m<-conmtrx(AY_A$stimulusWord, AY_A$selectedWord)
AZ_A.m<-conmtrx(AZ_A$stimulusWord, AZ_A$selectedWord)
```

Main analysis

For all subsequent figures and analysis, we'll use lexical set labels for the stimulus items.

```
LS_labels = c("TRAP", "BATH", "SQUARE ", "START","FACE","DRESS","FLEECE","NEAR","GOOSE","KIT","PRICE","
rownames(AA_A.m) = LS_labels
rownames(NN_N.m) = LS_labels
rownames(LL_L.m) = LS_labels
rownames(YY_Y.m) = LS_labels
rownames(ZZ_Z.m) = LS_labels

rownames(NA_N.m) = LS_labels
rownames(LA_L.m) = LS_labels
rownames(YA_Y.m) = LS_labels
rownames(ZA_Z.m) = LS_labels

rownames(AN_A.m) = LS_labels
rownames(AL_A.m) = LS_labels
rownames(AY_A.m) = LS_labels
rownames(AZ_A.m) = LS_labels
```

Calculate euclidean distance between vowel responses

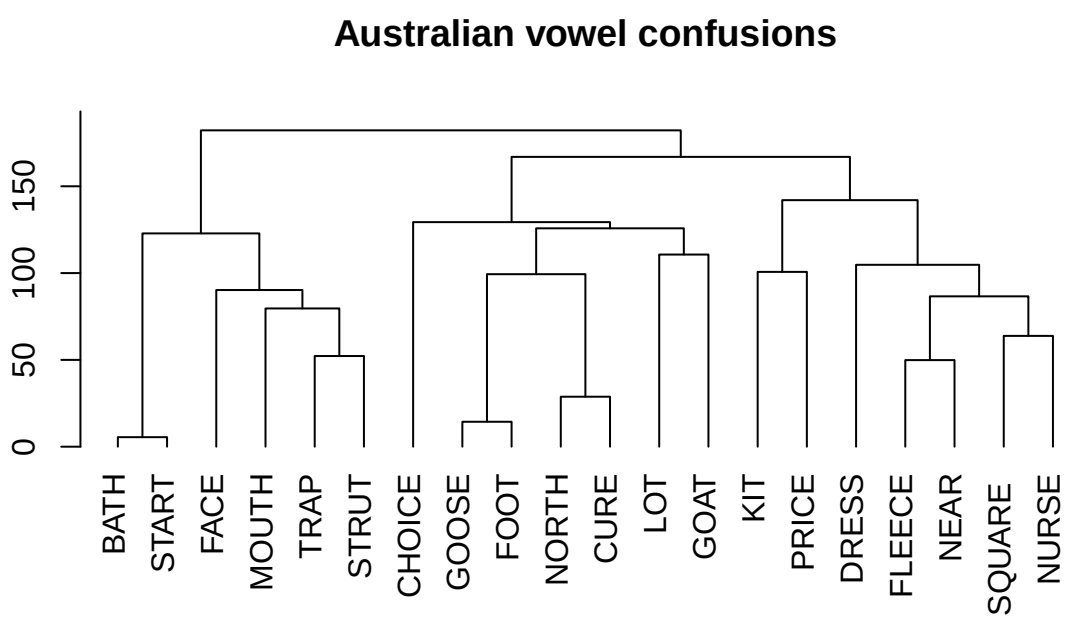
The distances between vowel response vectors are calculated as the square root of the sum of the squared differences in responses. See equation in numbered example (2) of the manuscript. In the resulting distance matrix, smaller numbers indicate greater similarity (less distance).

```
AA_A_Dist.m <- dist(AA_A.m)
NN_N_Dist.m<-dist(NN_N.m)
LL_L_Dist.m<-dist(LL_L.m)
YY_Y_Dist.m<-dist(YY_Y.m)
ZZ_Z_Dist.m<-dist(ZZ_Z.m)

NA_N_Dist.m<-dist(NA_N.m)
LA_L_Dist.m<-dist(LA_L.m)
YA_Y_Dist.m<-dist(YA_Y.m)
ZA_Z_Dist.m<-dist(ZA_Z.m)

AN_A_Dist.m<-dist(AN_A.m)
AL_A_Dist.m<-dist(AL_A.m)
AY_A_Dist.m<-dist(AY_A.m)
AZ_A_Dist.m<-dist(AZ_A.m)
```

Plot the distances between vowels

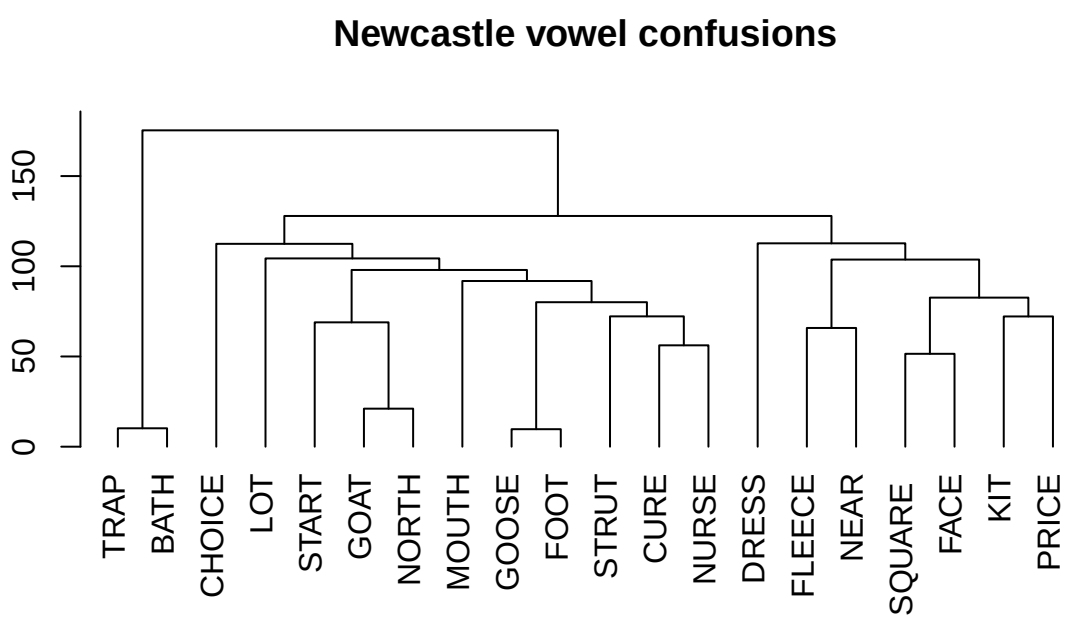
Create a hierarchical cluster object using Ward's minimum variance method, which finds compact, spherical clusters.Dendrogram for each accent.

```
AA_A_Dist.clus <-hclust(AA_A_Dist.m, method = "ward.D2")
NN_N_Dist.clus <-hclust(NN_N_Dist.m, method = "ward.D2")
LL_L_Dist.clus <-hclust(LL_L_Dist.m, method = "ward.D2")
YY_Y_Dist.clus <-hclust(YY_Y_Dist.m, method = "ward.D2")
ZZ_Z_Dist.clus <-hclust(ZZ_Z_Dist.m, method = "ward.D2")

plot(AA_A_Dist.clus, main="Australian vowel confusions", ylab="", axes=T, hang=-0.1)
```



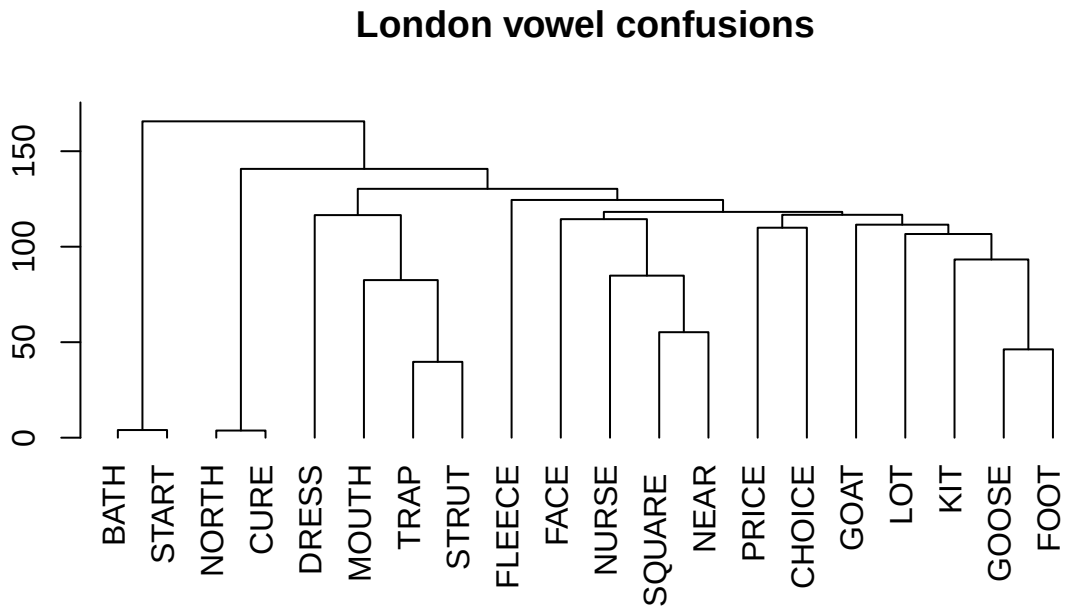
AA_A_Dist.m
hclust (*, "ward.D2")

```
plot(NN_N_Dist.clus, main="Newcastle vowel confusions", ylab="", axes=T, hang=-0.1)
```



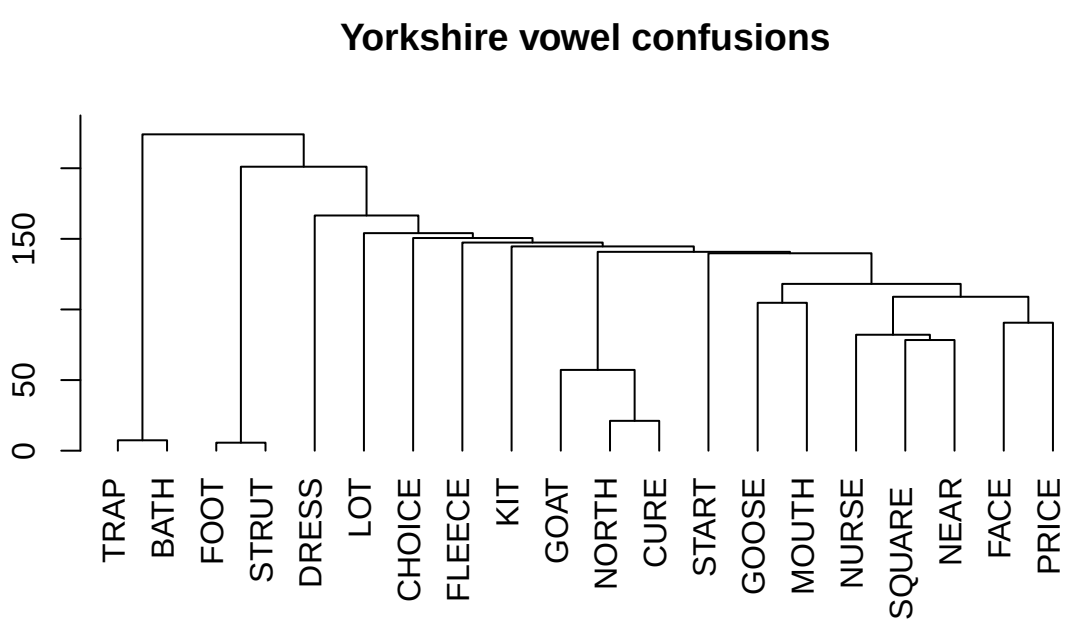
NN_N_Dist.m
hclust (*, "ward.D2")

```
plot(LL_L_Dist.clus, main="London vowel confusions", ylab="", axes=T, hang=-0.1)
```



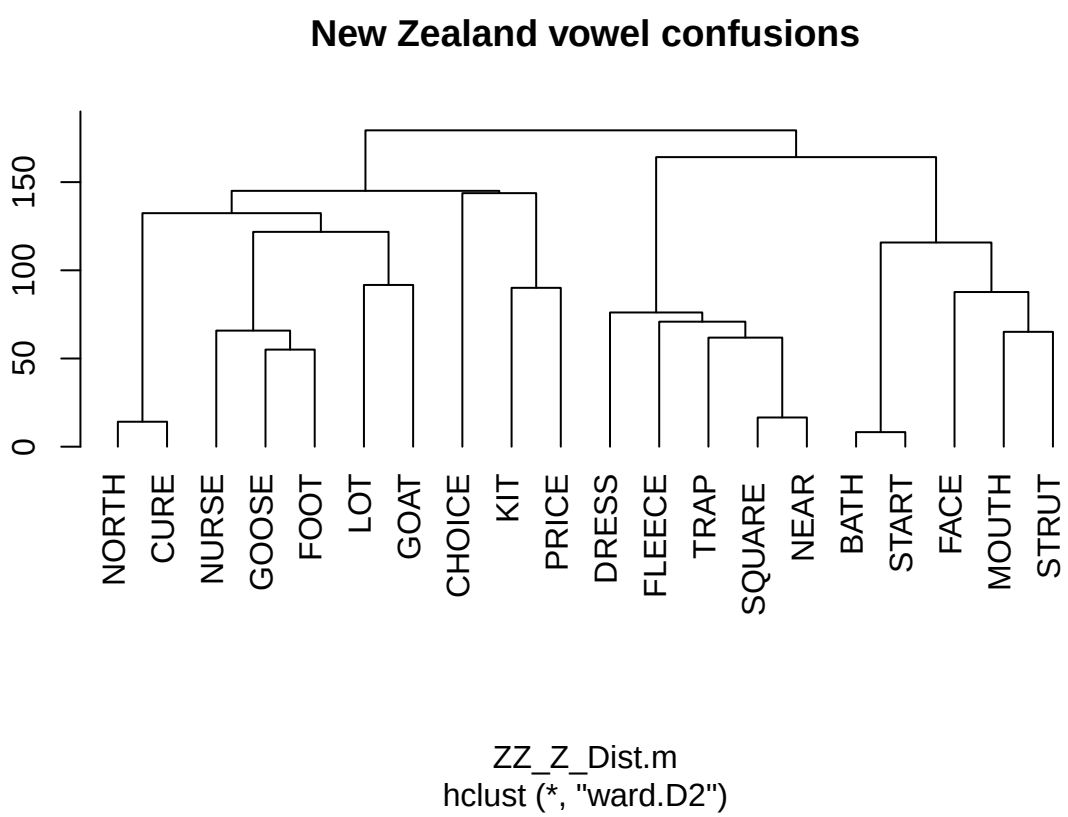
LL_L_Dist.m
hclust (*, "ward.D2")

```
plot(YY_Y_Dist.clus, main="Yorkshire vowel confusions", ylab="", axes=T, hang=-0.1)
```



YY_Y_Dist.m
hclust (*, "ward.D2")

```
plot(ZZ_Z_Dist.clus, main="New Zealand vowel confusions", ylab="", axes=T, hang=-0.1)
```



We next compute the distance between normalized vowel vectors across conditions

```
AA_An.m = AA_A.m/(sum(AA_A.m)/20)
ZA_Zn.m = ZA_Z.m/(sum(ZA_Z.m)/20)
LA_Ln.m = LA_L.m/(sum(LA_L.m)/20)
NA_Nn.m = NA_N.m/(sum(NA_N.m)/20)
YA_Yn.m = YA_Y.m/(sum(YA_Y.m)/20)

AL_An.m = AL_A.m/(sum(AL_A.m)/20)
AZ_An.m = AZ_A.m/(sum(AZ_A.m)/20)
AY_An.m = AY_A.m/(sum(AY_A.m)/20)
AN_An.m = AN_A.m/(sum(AN_A.m)/20)

ZZ_Zn.m = ZZ_Z.m/(sum(ZZ_Z.m)/20)
NN_Nn.m = NN_N.m/(sum(NN_N.m)/20)
LL_Ln.m = LL_L.m/(sum(LL_L.m)/20)
YY_Yn.m = YY_Y.m/(sum(YY_Y.m)/20)
```

Calculate Euclidean Distances between dialects for each normalized row.

```
words= list("zabba" , "zahba" , "zairba", "zarba" , "zayba" , "zebba" , "zeeba" , "zeerba", "zewba",  "z

meaned=0
sumed = 0
zaaa = 0
```

```
for (z in 1:length(words)) {
  ed = dist(rbind(AA_An.m[z,], ZA_Zn.m[z,]), method="euclidean")

  zaaa[z] = ed
  sumed = sumed + ed
}

words= list("zabba" , "zahba" , "zairba", "zarba" , "zayba" , "zebba" , "zeeba" , "zeerba", "zewba",  "z

meaned=0
sumed = 0
aazz=0
for (z in 1:length(words)) {
  ed = dist(rbind(AA_An.m[z,], ZZ_Zn.m[z,]), method="euclidean")

  sumed = sumed + ed
  aazz[z] = ed
}

words= list("zabba" , "zahba" , "zairba", "zarba" , "zayba" , "zebba" , "zeeba" , "zeerba", "zewba",  "z

meaned=0
sumed = 0
zazz=0
for (z in 1:length(words)) {
  ed = dist(rbind(ZA_Zn.m[z,], ZZ_Zn.m[z,]), method="euclidean")


  zazz[z] = ed
  sumed = sumed + ed
}

words= list("zabba" , "zahba" , "zairba", "zarba" , "zayba" , "zebba" , "zeeba" , "zeerba", "zewba",  "z

meaned=0
sumed = 0
yayy = 0
for (z in 1:length(words)) {
  ed = dist(rbind(YA_Yn.m[z,], YY_Yn.m[z,]), method="euclidean")


  yayy[z] = ed
  sumed = sumed + ed
}


words= list("zabba" , "zahba" , "zairba", "zarba" , "zayba" , "zebba" , "zeeba" , "zeerba", "zewba",  "z

meaned=0
sumed = 0
aayy = 0
for (z in 1:length(words)) {
  ed = dist(rbind(AA_An.m[z,], YY_Yn.m[z,]), method="euclidean")
```

```
  aayy[z] = ed
  sumed = sumed + ed
}


words= list("zabba" , "zahba" , "zairba", "zarba" , "zayba" , "zebba" , "zeeba" , "zeerba", "zewba",  "z

meaned=0
sumed = 0
yaaa = 0
for (z in 1:length(words)) {
  ed = dist(rbind(YA_Yn.m[z,], AA_An.m[z,]), method="euclidean")


  yaaa[z] = ed
  sumed = sumed + ed
}


words= list("zabba" , "zahba" , "zairba", "zarba" , "zayba" , "zebba" , "zeeba" , "zeerba", "zewba",  "z

meaned=0
sumed = 0
laaa = 0
for (z in 1:length(words)) {
  ed = dist(rbind(LA_Ln.m[z,], AA_An.m[z,]), method="euclidean")


  laaa[z] = ed
  sumed = sumed + ed
}


words= list("zabba" , "zahba" , "zairba", "zarba" , "zayba" , "zebba" , "zeeba" , "zeerba", "zewba",  "z

meaned=0
sumed = 0
aall = 0
for (z in 1:length(words)) {
  ed = dist(rbind(LL_Ln.m[z,], AA_An.m[z,]), method="euclidean")


  aall[z] = ed
  sumed = sumed + ed
}



words= list("zabba" , "zahba" , "zairba", "zarba" , "zayba" , "zebba" , "zeeba" , "zeerba", "zewba",  "z

meaned=0
sumed = 0
lall = 0
```

```
for (z in 1:length(words)) {
  ed = dist(rbind(LA_Ln.m[z,],LL_Ln.m[z,]), method="euclidean")


  lall[z] = ed
  sumed = sumed + ed
}

words= list("zabba" , "zahba" , "zairba", "zarba" , "zayba" , "zebba" , "zeeba" , "zeerba", "zewba",  "z

meaned=0
sumed = 0
naaa = 0
for (z in 1:length(words)) {
  ed = dist(rbind(NA_Nn.m[z,], AA_An.m[z,]), method="euclidean")


  naaa[z] = ed
  sumed = sumed + ed
}
words= list("zabba" , "zahba" , "zairba", "zarba" , "zayba" , "zebba" , "zeeba" , "zeerba", "zewba",  "z

meaned=0
sumed = 0
aann = 0
for (z in 1:length(words)) {
  ed = dist(rbind(NN_Nn.m[z,], AA_An.m[z,]), method="euclidean")


  aann[z] = ed
  sumed = sumed + ed
}

words= list("zabba" , "zahba" , "zairba", "zarba" , "zayba" , "zebba" , "zeeba" , "zeerba", "zewba",  "z

meaned=0
sumed = 0
nann= 0
for (z in 1:length(words)) {
  ed = dist(rbind(NA_Nn.m[z,], NN_Nn.m[z,]), method="euclidean")


  nann[z] = ed
  sumed = sumed + ed
}


words= list("zabba" , "zahba" , "zairba", "zarba" , "zayba" , "zebba" , "zeeba" , "zeerba", "zewba",  "z

meaned=0
sumed = 0
aaal= 0
for (z in 1:length(words)) {
```

```
  ed = dist(rbind(AA_An.m[z,], AL_An.m[z,]), method="euclidean")


  aaal[z] = ed
  sumed = sumed + ed
}

words= list("zabba" , "zahba" , "zairba", "zarba" , "zayba" , "zebba" , "zeeba" , "zeerba", "zewba",  "z

meaned=0
sumed = 0
aaaz= 0
for (z in 1:length(words)) {
  ed = dist(rbind(AA_An.m[z,], AZ_An.m[z,]), method="euclidean")


  aaaz[z] = ed
  sumed = sumed + ed
}

words= list("zabba" , "zahba" , "zairba", "zarba" , "zayba" , "zebba" , "zeeba" , "zeerba", "zewba",  "z

meaned=0
sumed = 0
aaay= 0
for (z in 1:length(words)) {
  ed = dist(rbind(AA_An.m[z,], AY_An.m[z,]), method="euclidean")


  aaay[z] = ed
  sumed = sumed + ed
}

words= list("zabba" , "zahba" , "zairba", "zarba" , "zayba" , "zebba" , "zeeba" , "zeerba", "zewba",  "z

meaned=0
sumed = 0
aaan= 0
for (z in 1:length(words)) {
  ed = dist(rbind(AA_An.m[z,], AN_An.m[z,]), method="euclidean")


  aaan[z] = ed
  sumed = sumed + ed
}
```

```
words= list("zabba" , "zahba" , "zairba", "zarba" , "zayba" , "zebba" , "zeeba" , "zeerba", "zewba",  "z

meaned=0
sumed = 0
nnll= 0
for (z in 1:length(words)) {
  ed = dist(rbind(NN_Nn.m[z,], LL_Ln.m[z,]), method="euclidean")
```

```
  nnll[z] = ed
  sumed = sumed + ed
}

words= list("zabba" , "zahba" , "zairba", "zarba" , "zayba" , "zebba" , "zeeba" , "zeerba", "zewba",  "z

meaned=0
sumed = 0
yyll= 0
for (z in 1:length(words)) {
  ed = dist(rbind(YY_Yn.m[z,], LL_Ln.m[z,]), method="euclidean")


  yyll[z] = ed
  sumed = sumed + ed
}

words= list("zabba" , "zahba" , "zairba", "zarba" , "zayba" , "zebba" , "zeeba" , "zeerba", "zewba",  "z

meaned=0
sumed = 0
nnzz= 0
for (z in 1:length(words)) {
  ed = dist(rbind(NN_Nn.m[z,], ZZ_Zn.m[z,]), method="euclidean")


  nnzz[z] = ed
  sumed = sumed + ed
}

words= list("zabba" , "zahba" , "zairba", "zarba" , "zayba" , "zebba" , "zeeba" , "zeerba", "zewba",  "z

meaned=0
sumed = 0
yyzz= 0
for (z in 1:length(words)) {
  ed = dist(rbind(YY_Yn.m[z,], ZZ_Zn.m[z,]), method="euclidean")


  yyzz[z] = ed
  sumed = sumed + ed
}

words= list("zabba" , "zahba" , "zairba", "zarba" , "zayba" , "zebba" , "zeeba" , "zeerba", "zewba",  "z

meaned=0
sumed = 0
llzz= 0
for (z in 1:length(words)) {
  ed = dist(rbind(LL_Ln.m[z,], ZZ_Zn.m[z,]), method="euclidean")
```

```
  llzz[z] = ed
  sumed = sumed + ed
}

meaned=0
sumed = 0
nnyy= 0
for (z in 1:length(words)) {
  ed = dist(rbind(NN_Nn.m[z,], YY_Yn.m[z,]), method="euclidean")


  nnyy[z] = ed
  sumed = sumed + ed
}


meaned=0
sumed = 0
alll= 0
for (z in 1:length(words)) {
  ed = dist(rbind(AL_An.m[z,], LL_Ln.m[z,]), method="euclidean")


  alll[z] = ed
  sumed = sumed + ed
}

meaned=0
sumed = 0
azzz= 0
for (z in 1:length(words)) {
  ed = dist(rbind(AZ_An.m[z,], ZZ_Zn.m[z,]), method="euclidean")


  azzz[z] = ed
  sumed = sumed + ed
}

meaned=0
sumed = 0
ayyy= 0
for (z in 1:length(words)) {
  ed = dist(rbind(AY_An.m[z,], YY_Yn.m[z,]), method="euclidean")


  ayyy[z] = ed
  sumed = sumed + ed
}

meaned=0
sumed = 0
annn= 0
for (z in 1:length(words)) {
```

```
  ed = dist(rbind(AN_An.m[z,], NN_Nn.m[z,]), method="euclidean")


  annn[z] = ed
  sumed = sumed + ed
}
```

Next we'll add up the cross-accent differences. The sum is divided by the theoretical maximum to yield a similarity scale that varies from 0 to 1; see equation in (5) of the manuscript. These values are reported in Figure 3f of the manuscript and re-occur for reference in Tables 4 and 5.

```
# normalize by the maximum possible difference to place the differences on a scale

aall_EDn <- sum(aall)/(sqrt(2)*20); print(aall_EDn)
```

```
## [1] 0.2062243
```

```
aazz_EDn <- sum(aazz)/(sqrt(2)*20); print(aazz_EDn)
```

```
## [1] 0.1208941
```

```
aayy_EDn <- sum(aayy)/(sqrt(2)*20); print(aayy_EDn)
```

```
## [1] 0.2231988
```

```
aann_EDn <- sum(aann)/(sqrt(2)*20); print(aann_EDn)
```

```
## [1] 0.2172081
```

```
llzz_EDn <- sum(llzz)/(sqrt(2)*20); print(llzz_EDn)
```

```
## [1] 0.2140189
```

```
yyll_EDn <- sum(yyll)/(sqrt(2)*20); print(yyll_EDn)
```

```
## [1] 0.224937
```

```
nnll_EDn <- sum(nnll)/(sqrt(2)*20); print(nnll_EDn)
```

```
## [1] 0.2712878
```

```
yyzz_EDn <- sum(yyzz)/(sqrt(2)*20); print(yyzz_EDn)
```

```
## [1] 0.2358074
```

```
nnzz_EDn <- sum(nnzz)/(sqrt(2)*20); print(nnzz_EDn)
```

```
## [1] 0.2487961
```

```
nnyy_EDn <- sum(nnyy)/(sqrt(2)*20); print(nnyy_EDn)
```

```
## [1] 0.1683938
```

Table 4 values: other accent listeners of Australian accent vowels

```
# normalize by the maximum possible difference to place the differences on a scale

aall_EDn <- sum(aall)/(sqrt(2)*20)
aazz_EDn <- sum(aazz)/(sqrt(2)*20)
aayy_EDn <- sum(aayy)/(sqrt(2)*20)
aann_EDn <- sum(aann)/(sqrt(2)*20)
print(aall_EDn)
```

```
## [1] 0.2062243
```

```
print(aazz_EDn)
```

```
## [1] 0.1208941
```

```
print(aayy_EDn)
```

```
## [1] 0.2231988
```

```
print(aann_EDn)
```

```
## [1] 0.2172081
```

```
aala_EDn <- sum(laaa)/(sqrt(2)*20)
aaza_EDn <- sum(zaaa)/(sqrt(2)*20)
aaya_EDn <- sum(yaaa)/(sqrt(2)*20)
aana_EDn <- sum(naaa)/(sqrt(2)*20)
print(aala_EDn)
```

```
## [1] 0.1658676
```

```
print(aaza_EDn)
```

```
## [1] 0.1968623
```

```
print(aaya_EDn)
```

```
## [1] 0.1732689
```

```
print(aana_EDn)
```

```
## [1] 0.208418
```

```
lall_EDn <- sum(lall)/(sqrt(2)*20)
zazz_EDn <- sum(zazz)/(sqrt(2)*20)
yayy_EDn <- sum(yayy)/(sqrt(2)*20)
nann_EDn <- sum(nann)/(sqrt(2)*20)
print(lall_EDn)
```

```
## [1] 0.1578167
```

```
print(zazz_EDn)
```

```
## [1] 0.2115246
```

```
print(yayy_EDn)
```

```
## [1] 0.2457894
```

```
print(nann_EDn)
```

```
## [1] 0.2731348
```

Table 5 values; Australian listeners of other accent vowels

```
# normalize by the maximum possible difference to place the differences on a scale

aall_EDn <- sum(aall)/(sqrt(2)*20)
aazz_EDn <- sum(aazz)/(sqrt(2)*20)
aayy_EDn <- sum(aayy)/(sqrt(2)*20)
aann_EDn <- sum(aann)/(sqrt(2)*20)
print(aall_EDn)
```

```
## [1] 0.2062243
```

```
print(aazz_EDn)
```

```
## [1] 0.1208941
```

```
print(aayy_EDn)
```

```
## [1] 0.2231988
```

```
print(aann_EDn)
```

```
## [1] 0.2172081
```

```
aaal_EDn <- sum(aaal)/(sqrt(2)*20)
aaaz_EDn <- sum(aaaz)/(sqrt(2)*20)
aaay_EDn <- sum(aaay)/(sqrt(2)*20)
aaan_EDn <- sum(aaan)/(sqrt(2)*20)
print(aaal_EDn)
```

```
## [1] 0.1305933
```

```
print(aaaz_EDn)
```

```
## [1] 0.1741424
```

```
print(aaay_EDn)
```

```
## [1] 0.1722899
```

```
print(aaan_EDn)
```

```
## [1] 0.2168827
```

```
# control comparisons
alll_EDn <- sum(alll)/(sqrt(2)*20)
azzz_EDn <- sum(azzz)/(sqrt(2)*20)
ayyy_EDn <- sum(ayyy)/(sqrt(2)*20)
annn_EDn <- sum(annn)/(sqrt(2)*20)
print(alll_EDn)
```

```
## [1] 0.2005281
```

```
print(azzz_EDn)
```

```
## [1] 0.1434233
```

```
print(ayyy_EDn)
```

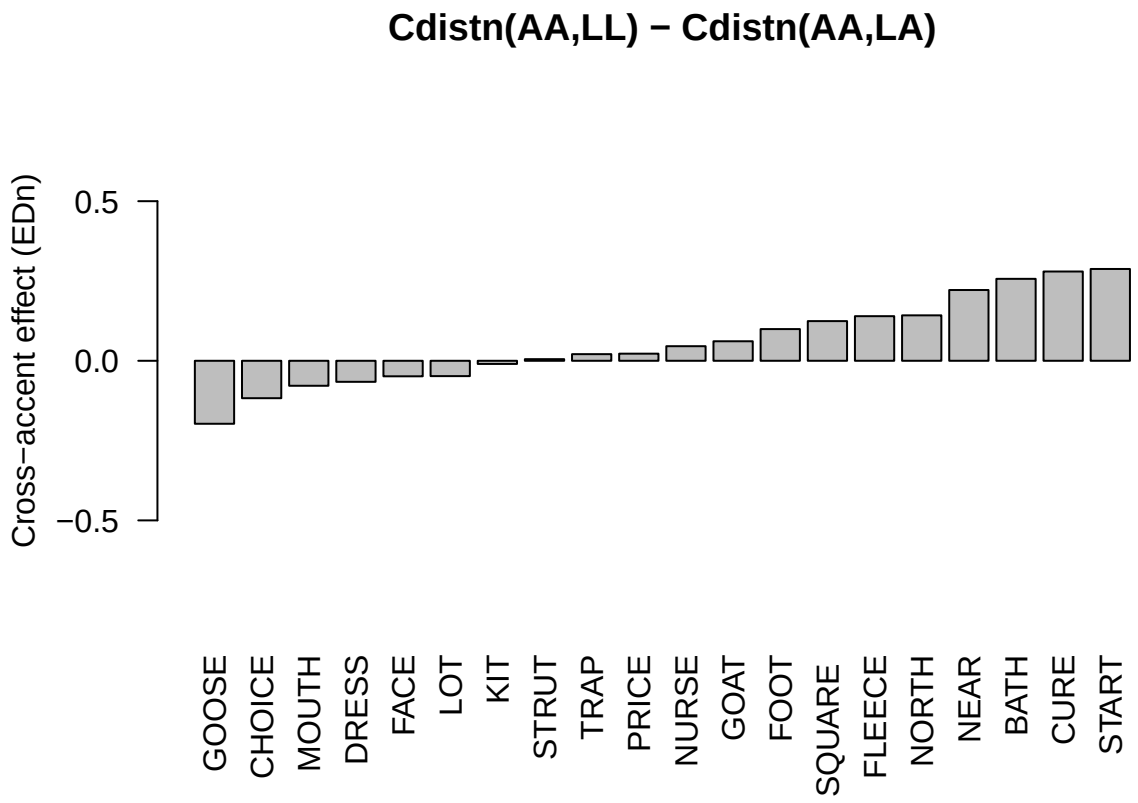
```
## [1] 0.2462201
```

```
print(annn_EDn)
```

```
## [1] 0.2340652
```

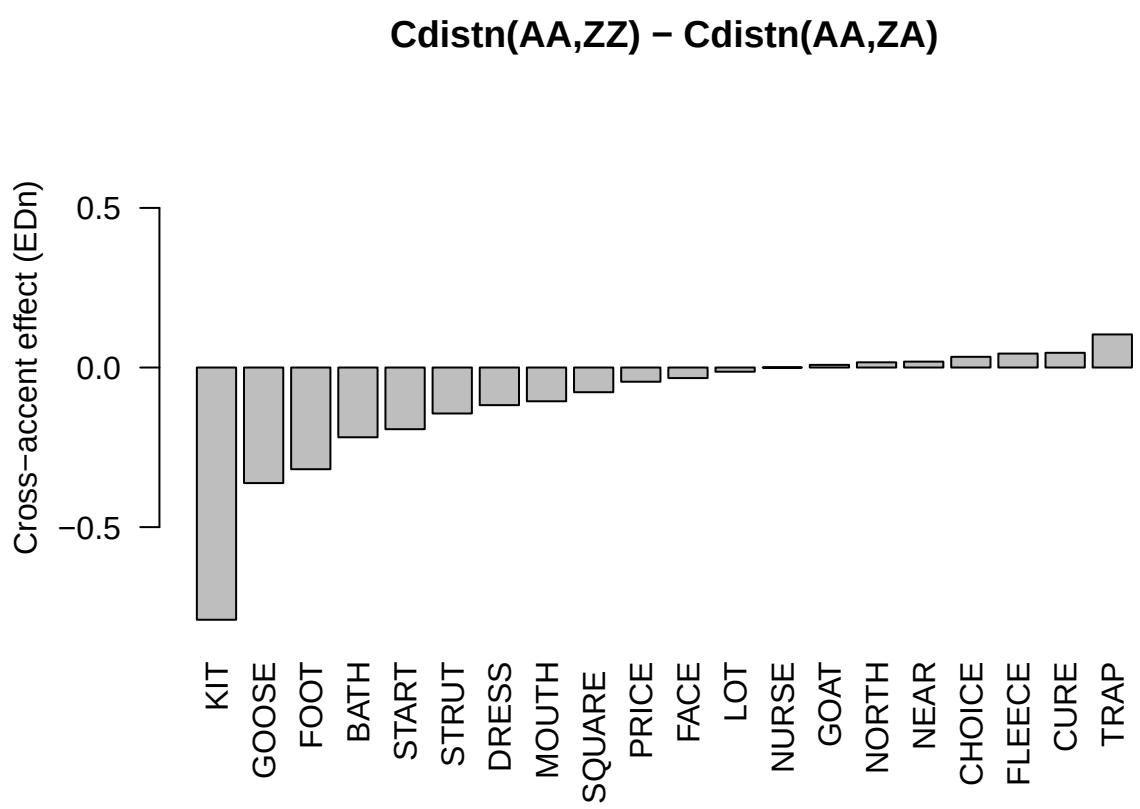
Figure 4: Which vowels contribute most to the cross-accent results? Other listeners of Australian accent vowels.

```
# Australian listeners London vowels
cae_la <- aall-laaa
LS_labels <- as.data.frame(LS_labels)
cae_la <- cbind(cae_la,LS_labels)
cae_las <- cae_la[order(cae_la$cae_la),]

# barchart of differences
barplot(cae_las[,1],
main = "Cdistn(AA,LL) - Cdistn(AA,LA)",
ylab = "Cross-accent effect (EDn)",
ylim= c(-0.8,0.8),
names.arg = cae_las[,2],
horiz = FALSE,
las = 2)
```

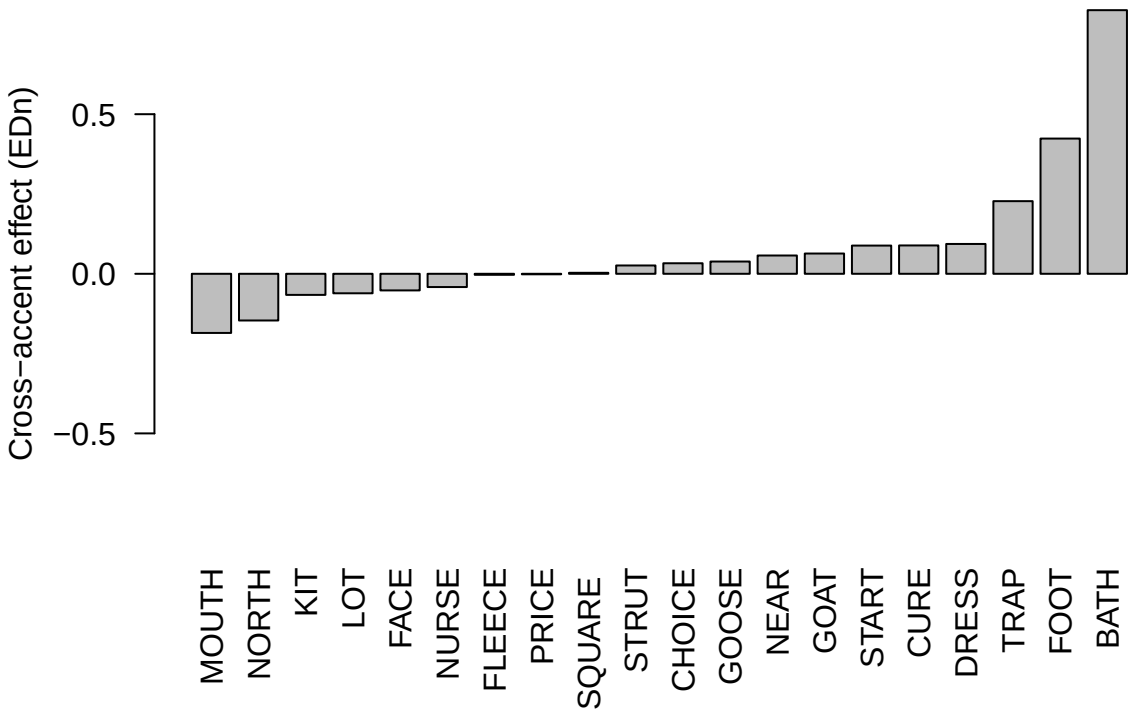



```
# Australian listeners New Zealand vowels
cae_za <- aazz-zaaa
LS_labels <- as.data.frame(LS_labels)
cae_za <- cbind(cae_za,LS_labels)
cae_zas <- cae_za[order(cae_za$cae_za),]

# barchart of differences
barplot(cae_zas[,1],
main = "Cdistn(AA,ZZ) - Cdistn(AA,ZA)",
ylab = "Cross-accent effect (EDn)",
```

```
ylim= c(-0.8,0.8),
names.arg = cae_zas[,2],
horiz = FALSE,
las = 2)
```




```
# Australian listeners Yorkshire vowels
cae_ya <- aayy-yaaa
LS_labels <- as.data.frame(LS_labels)
cae_ya <- cbind(cae_ya,LS_labels)
cae_yas <- cae_ya[order(cae_ya$cae_ya),]

# barchart of differences
barplot(cae_yas[,1],
main = "Cdistn(AA,YY) - Cdistn(AA,YA)",
ylab = "Cross-accent effect (EDn)",
ylim= c(-0.8,0.8),
names.arg = cae_yas[,2],
horiz = FALSE,
las = 2)
```




```
# Australian listeners Newcastle vowels
cae_na <- aann-naaa
LS_labels <- as.data.frame(LS_labels)
cae_na <- cbind(cae_na,LS_labels)
cae_nas <- cae_na[order(cae_na$cae_na),]

# barchart of differences
barplot(cae_nas[,1],
main = "Cdistn(AA,NN) - Cdistn(AA,NA)",
ylab = "Cross-accent effect (EDn)",
ylim= c(-0.8,0.8),
names.arg = cae_nas[,2],
horiz = FALSE,
las = 2
)
```

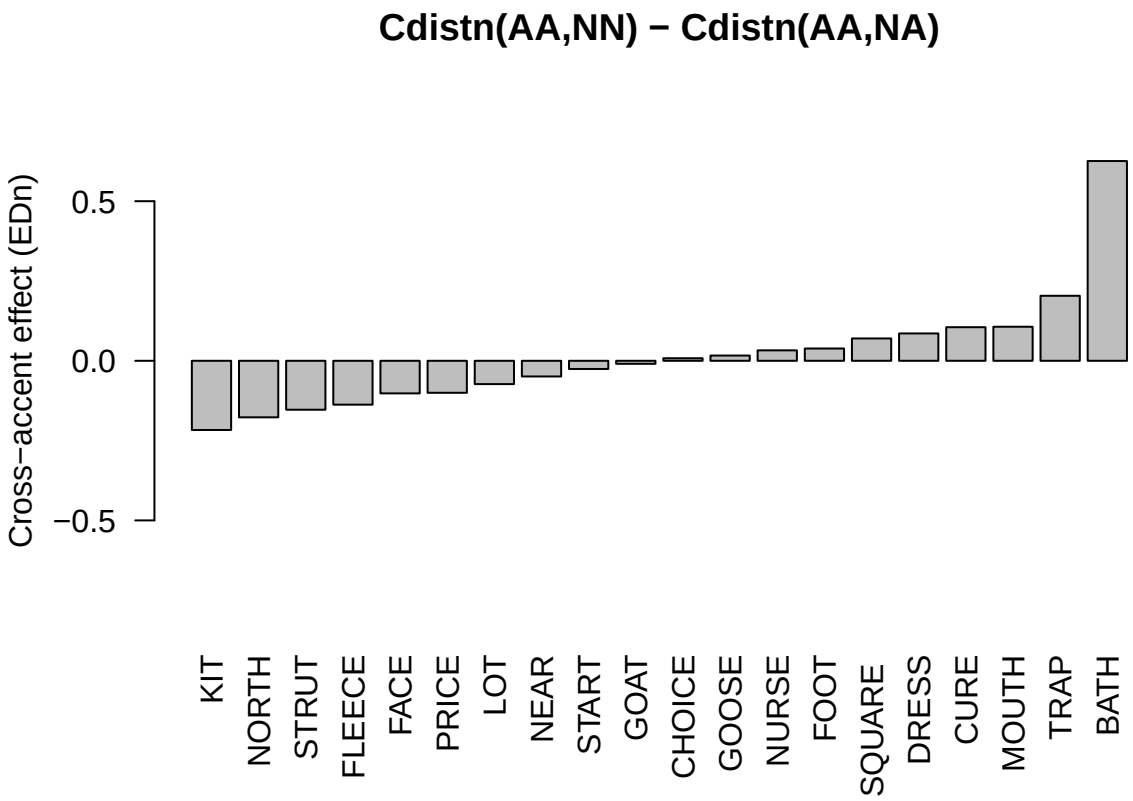
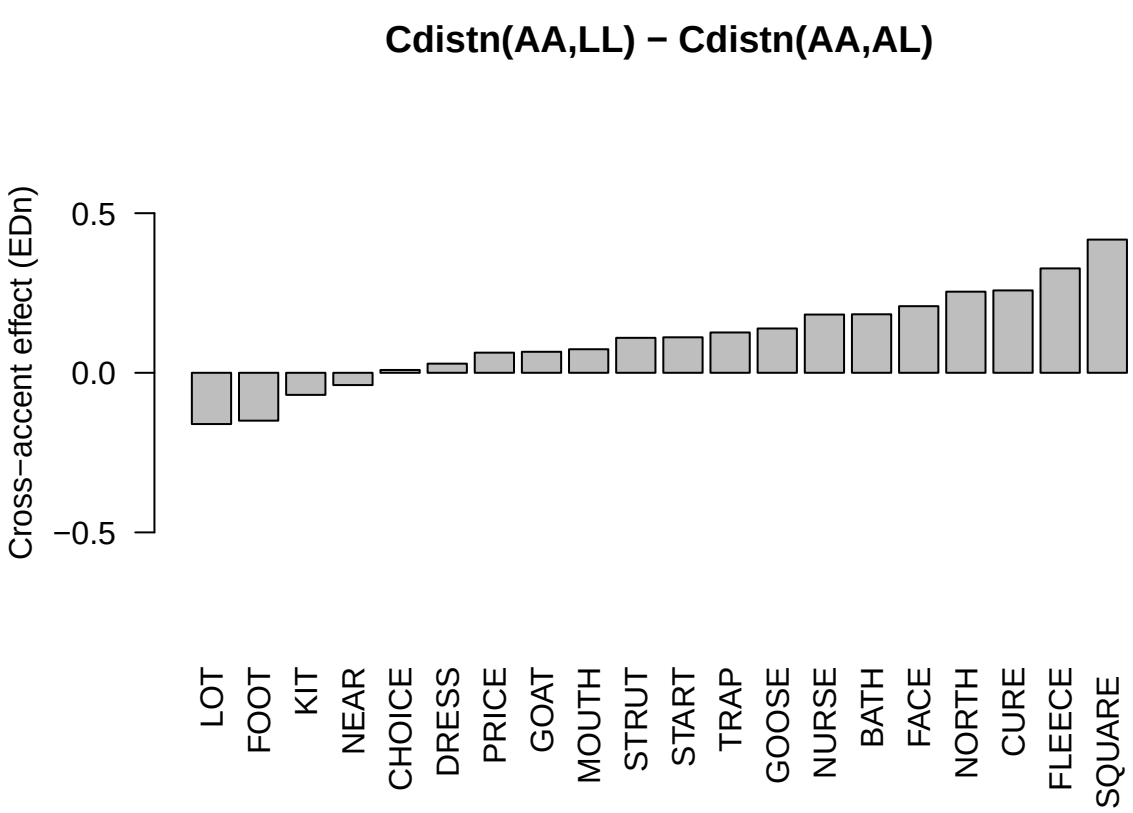
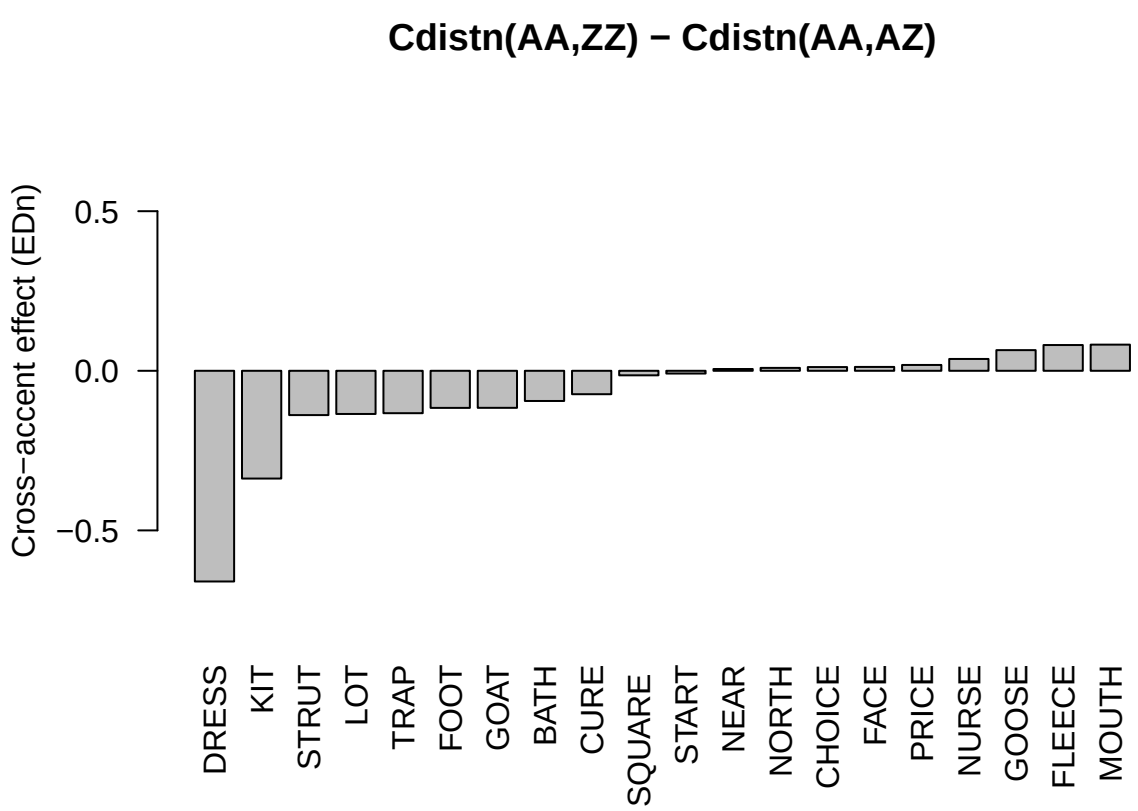


Figure 5: Which vowels contribute most to the cross-accent results? Australian listeners of other accents.

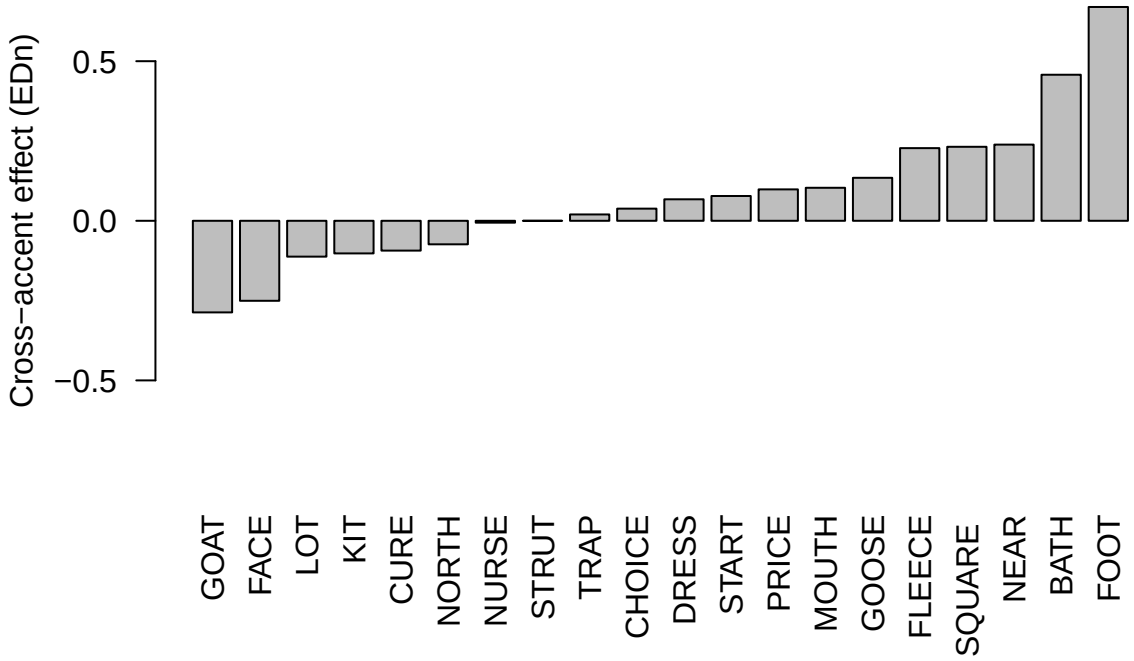
```
# Australian listeners London vowels
cae_al <- aall-aaal
LS_labels <- as.data.frame(LS_labels)
cae_al <- cbind(cae_al,LS_labels)
cae_als <- cae_al[order(cae_al$cae_al),]

# barchart of differences
barplot(cae_als[,1],
main = "Cdistn(AA,LL) - Cdistn(AA,AL)",
ylab = "Cross-accent effect (EDn)",
ylim= c(-0.8,0.8),
names.arg = cae_als[,2],
horiz = FALSE,
las = 2)
```



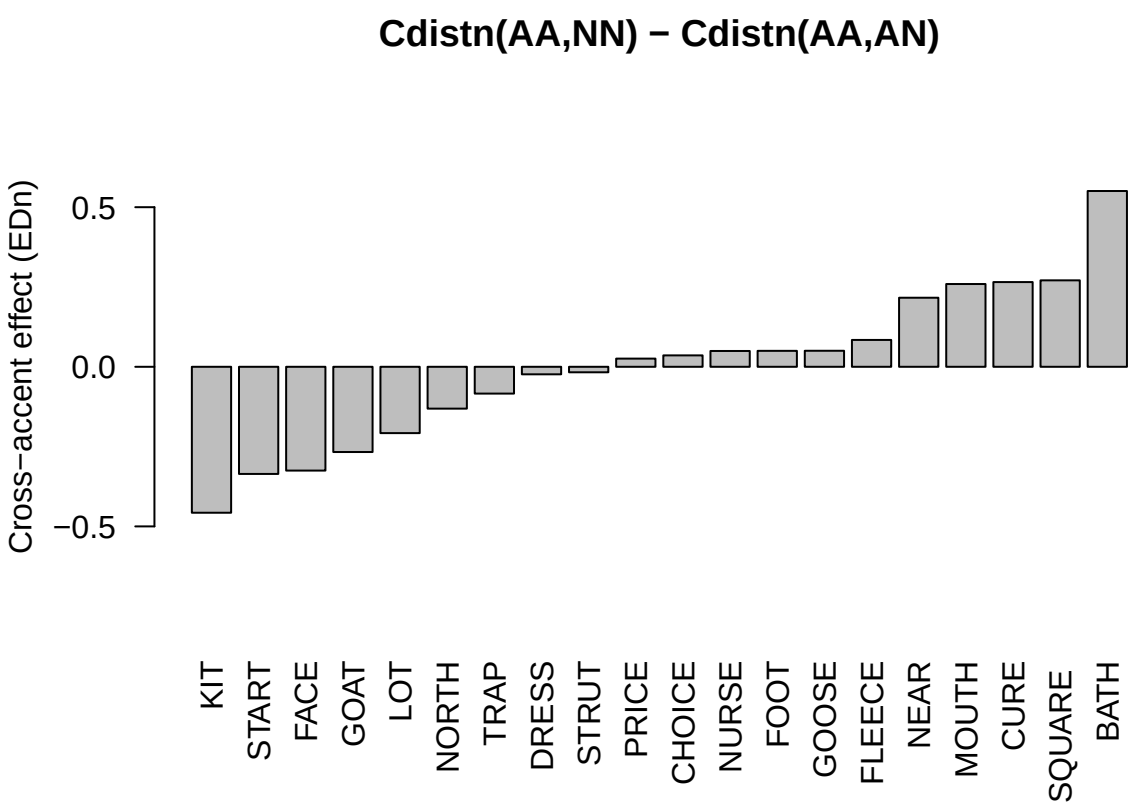

```
# Australian listeners New Zealand vowels
cae_az <- aazz-aaaz
LS_labels <- as.data.frame(LS_labels)
cae_az <- cbind(cae_az,LS_labels)
cae_azs <- cae_az[order(cae_az$cae_az),]

# barchart of differences
barplot(cae_azs[,1],
main = "Cdistn(AA,ZZ) - Cdistn(AA,AZ)",
ylab = "Cross-accent effect (EDn)",
ylim= c(-0.8,0.8),
names.arg = cae_azs[,2],
horiz = FALSE,
las = 2)
```




```
# Australian listeners Yorkshire vowels
cae_ay <- aayy-aaay
LS_labels <- as.data.frame(LS_labels)
cae_ay <- cbind(cae_ay,LS_labels)
cae_ays <- cae_ay[order(cae_ay$cae_ay),]

# barchart of differences
barplot(cae_ays[,1],
main = "Cdistn(AA,YY) - Cdistn(AA,AY)",
ylab = "Cross-accent effect (EDn)",
ylim= c(-0.8,0.8),
names.arg = cae_ays[,2],
horiz = FALSE,
las = 2)
```




```
# Australian listeners Newcastle vowels
cae_an <- aann-aaan
LS_labels <- as.data.frame(LS_labels)
cae_an <- cbind(cae_an,LS_labels)
cae_ans <- cae_an[order(cae_an$cae_an),]

# barchart of differences
barplot(cae_ans[,1],
main = "Cdistn(AA,NN) - Cdistn(AA,AN)",
ylab = "Cross-accent effect (EDn)",
ylim= c(-0.8,0.8),
names.arg = cae_ans[,2],
horiz = FALSE,
las = 2)
```



Negative numbers indicate that perceptual distance increases for that vowel in the cross-accent scenario (that is, these vowels are more confusing in the cross-accent scenario); positive numbers indicate that perceptual distance decreases (these vowels are less confusing/more like native listener) in the cross-accent scenario. Lower perceptual distance in the cross-accent scenario (perceptual assimilation) is driven by the vowels with the largest positive numbers (right side of figure) LONDON: negative effect (perceptual distance increases for) for LOT, FOOT, KIT, NEAR; positive effect for 15 vowels; largest for SQUARE and then FLEECE.

New Zealand: majority negative effects, largest negative effects are for DRESS then KIT

Yorkshire: largest negative effects are for GOAT and FACE; largest positive effects are for BATH and FOOT

Newcastle: largest negative effect for KIT (also START, FACE, GOAT); largest positive effect is for BATH