Get freqs for NWR YD project

AC

9/8/2020

Initial clean up

This is a better than what we had in divime, because it does more cleanup. To run it, you need to be in raw_YEL

```
grep "^[FM]A" txt/*.txt | # use only adult speech
cut -f 6 | #take transcription
sed "s/\[: /\[:_/g" |  sed "s/\[=! /\[:_/g" | sed "s/\[- /\[-_/g" | #remove spaces that are not real on
grep -v "\[-_" | #remove sentences in Eng and Pidgin
 grep -v "he " | grep -vw "they" | grep -vw "I"  | grep -vw "you" | grep -vw "your"| grep -vw "his" | g
grep -vw "alright" |grep -vw "already" |grep -vw "almost" |grep -vw "another" |
grep -vw "bye" |grep -vw "calling" |grep -vw "coming" |grep -vw "clothes" |
grep -vw "complete" |grep -vw "counting" |grep -vw "enough" |grep -vw "fight" |
grep -vw "first" |grep -vw "finished" |grep -vw "there"|grep -vw "which" |grep -vw "witchcraft" |grep -
grep -vw "window" |grep -vw "want"|grep -vw "sister"|grep -vw "sleeping"|
grep -vw "sometimes" | grep -vw "somehwere"|grep -vw "scared"|grep -vw "saw" |
grep -vw "inside" | grep -vw "gonna" | grep -vw "her" | grep -vw "him"|
grep -vw "here" | grep -vw "easy" | grep -vw "early" | grep -vw "eye"|
grep -vw "cook" | grep -vw "dog" | grep -vw "area" | grep -vw "around"|
grep -v "a@l" | grep -v "e@l" | grep -v "d@l"|
sed 's/\[[^[]*\]//g' | #delete comments
tr ' ' '\n' | #cut at word boundaries
tr -d '?' | tr -d '.' | tr -d '!' | tr -d '-' | tr -d ',' | #clean up punctuation
grep -v "&" | grep -v "@s" | #remove switches
grep -v "xxx" | grep -vw "xx" | grep -vw "hm" | grep -vw "mm" | grep -vw "mmhm" | grep -vw "oh" | grep
grep -v "[A-Z]" | grep -vw "Ńaamońo" | #get rid of all names
sed "s/aa+/aa/g" | sed "s/ee+/ee/g" | sed "s/ii+/ii/g" | sed "s/oo+/oo/g" | sed "s/uu+/aa/g" |
sed "s/êê+/êê/g" | sed "s/ââ+/ââ/g" | sed "s/áá+/áá/g" | sed "s/óó+/óó/g" | #exaggeration of vowel leng
sed "s/aaaa/aa/g" |
tr -d '>' | tr -d '<' |tr -d '(' |tr -d ')' | sed "s/@c//g" | # final cleaning and write out
sed "s/che/te/g" |  sed "s/chi/ti/g" | sort | grep -v "[0-9]" | sed '/^$/d' > words_corpus.txt
```

Getting frequencies for segments in our stimuli

Middy gave me an onset list that converts all vowels and consonants to common representations for the purposes of counting syllables (rossel-ortho-replacements.txt). I thought of the following changes:

- remove all rewrites for vowels
- but that destroys the context for the following rules, eg not removing colon means knw does not find a match in knw:a
- but why weren't these defined with regular expressions anyway?

- it looks like I should be careful with the 4- and 3-letter phonemes, because I do not have those in my rewrites but not otherwise

So all things considered, it looks like it may be easier to just do the replacement of the 4- and 3-letter segments here, just being careful to map these to something that is not used in the stimuli, such as ngm.

Also, this reveals my rewrites were incomplete, so I added some lines for the onsets that were missing in my initial correspondance list.

The correspondances here looks similar to the one in wrangling but it has more entries because we want to separate each phoneme, and we want to distinguish between short and long

```
# get frequencies in raw_YEL

scan("words_corpus.txt",what="char")->wds

#wds[1:1000]

#initialize the unichar phono-like representation
wds_uni=wds

for(i in 1:dim(correspondances)[1]) { #transform into unicharacter
  wds_uni=gsub(correspondances[i,1],correspondances[i,2],wds_uni,fixed=T,useBytes = T)
}
#wds_uni[1:1000]

wds_uni=unlist(strsplit(wds_uni,split=" "))
counts=data.frame(table(wds_uni))
counts$wds_uni=as.character(counts$wds_uni)

correspondances2=correspondances
colnames(correspondances2)<-c("ortho","fake")
correspondances2=data.frame(correspondances2)
correspondances2$fake=gsub(" ","",as.character(correspondances2$fake))

#backtranslate
merge(counts,correspondances2,by.x="wds_uni",by.y="fake")->counts
counts=counts[order(counts$Freq),]

colnames(counts)[2]<-"counts_corpus"
counts$freq_corpus=counts$counts_corpus/sum(counts$counts_corpus)

write.table(counts,"segment-counts.txt",col.names = T,row.names = F,quote=T,sep="\t")
```

```
#initialize the unichar phono-like representation
wds_uni= names(table(wds))

for(i in 1:dim(correspondances)[1]) { #transform into unicharacter
  wds_uni=gsub(correspondances[i,1],correspondances[i,2],wds_uni,fixed=T,useBytes = T)
}
#wds_uni[1:1000]

wds_uni=unlist(strsplit(wds_uni,split=" "))
counts=data.frame(table(wds_uni))
counts$wds_uni=as.character(counts$wds_uni)
```

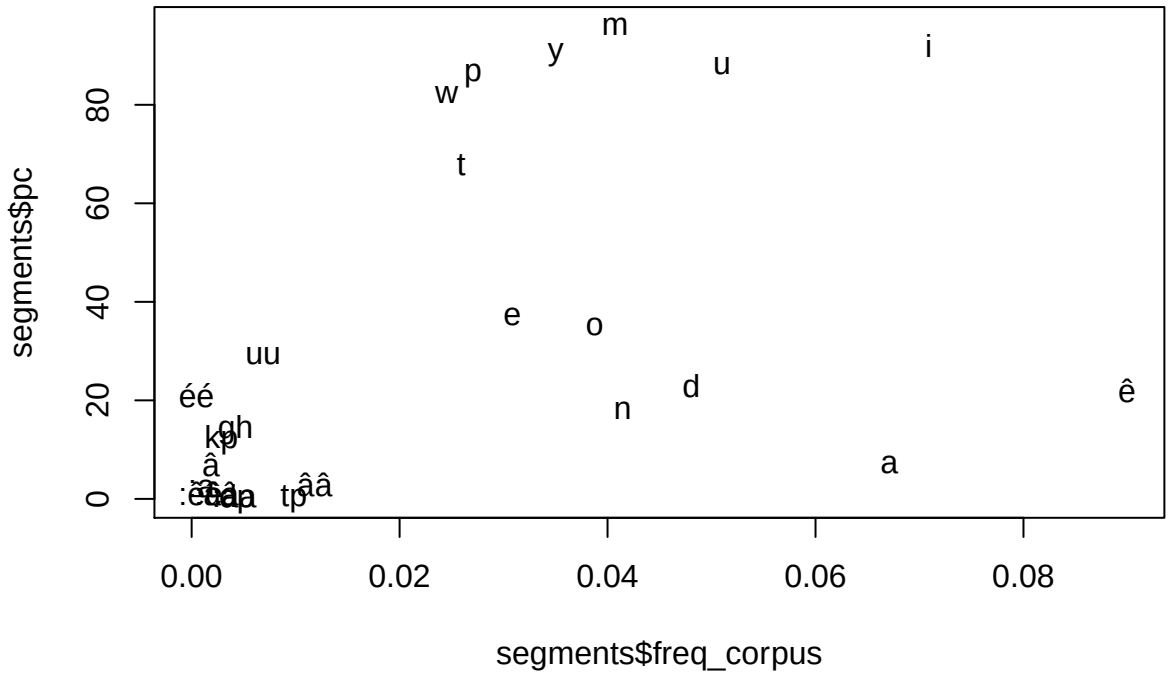
```
colnames(correspondances)<-c("ortho","fake")
correspondances=data.frame(correspondances)
correspondances$fake=gsub(" ","",as.character(correspondances$fake))

#backtranslate
merge(counts,correspondances,by.x="wds_uni",by.y="fake")->counts
counts=counts[order(counts$Freq),]

colnames(counts)[2]<-"counts_corpus_types"
counts$freq_corpus_types=counts$counts_corpus/sum(counts$counts_corpus)

write.table(counts,"segment-counts-types.txt",col.names = T,row.names = F,quote=T,sep="\t")
```

Analyses

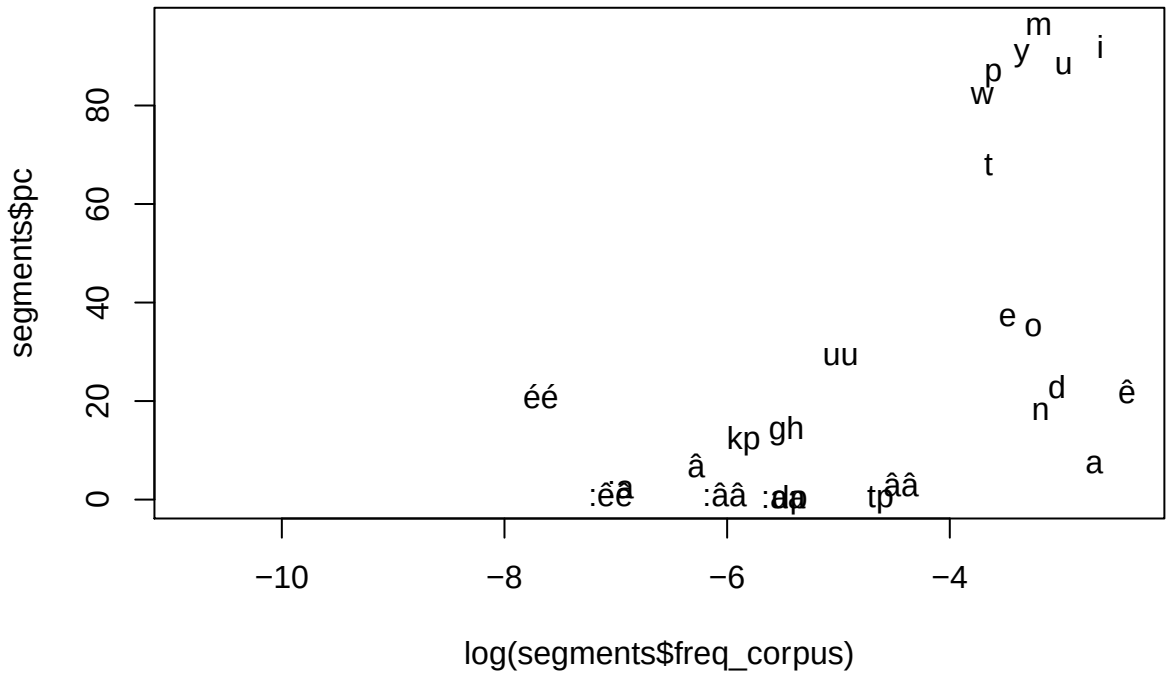
I ran the code in wrangling again, to integrate these counts & freqs into the main data set.

In addition, I wrote out a new version of segments which has the corpus frequency

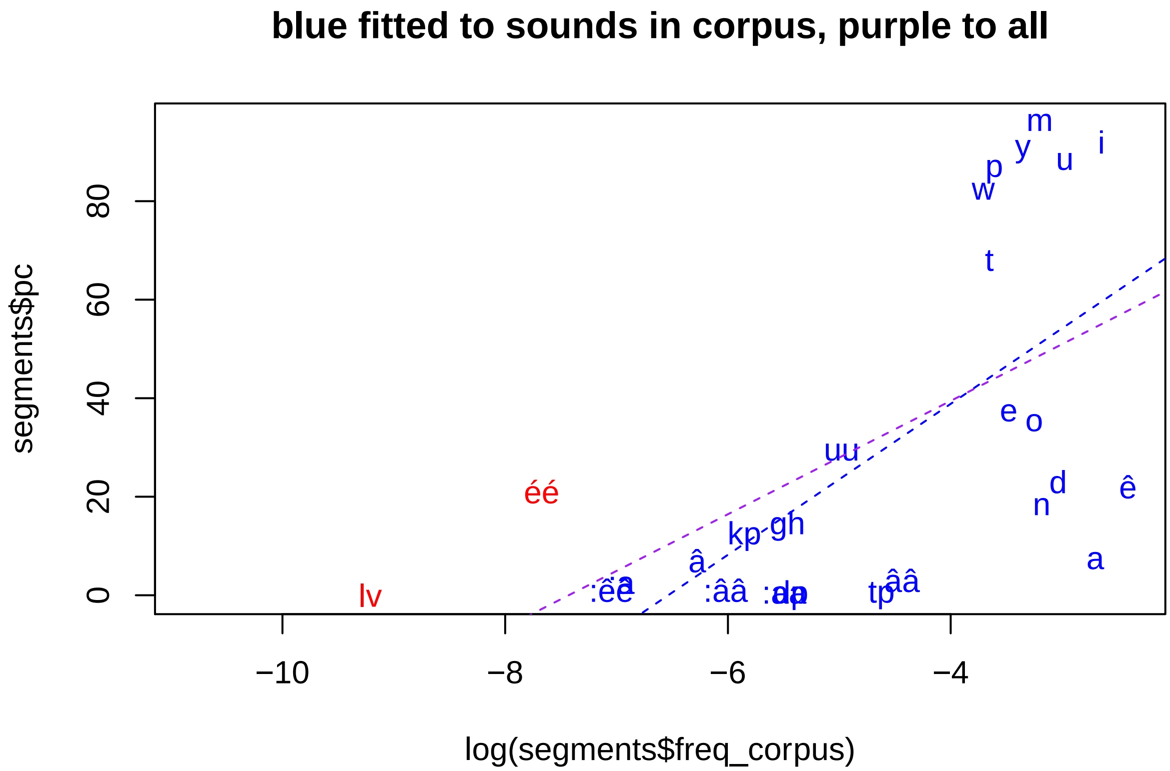
```
read.table("segments_with_cor_freq.txt",header=T)->segments # not ran
plot(segments$pc~segments$freq_corpus,type="n")
text(segments$pc~segments$freq_corpus,labels=segments$ortho)
```




```
plot(segments$pc~log(segments$freq_corpus),type="n")
text(segments$pc~log(segments$freq_corpus),labels=segments$ortho)
```




```
#some stimuli sounds do not occur in the corpus at all
#so we'll give them a really small frequency, just so that they show up
#and we tag them in red
segments$freq_corpus[is.na(segments$freq_corpus)]<-.0001
plot(segments$pc~log(segments$freq_corpus),type="n",main="blue fitted to sounds in corpus, purple to al
text(segments$pc~log(segments$freq_corpus),labels=segments$ortho,col=ifelse(segments$freq_corpus<.00057
#abline(lm(segments$pc~log(segments$freq_corpus),subset=c(segments$freq_corpus<.00057)),col="red",lty=2
abline(lm(segments$pc~log(segments$freq_corpus),subset=c(segments$freq_corpus>=.00057)),col="blue",lty=
abline(lm(segments$pc~log(segments$freq_corpus)),col="purple",lty=2)
```


blue fitted to sounds in corpus, purple to all
segments$pc
log(segments$freq_corpus)
0
20
40
60
80
−10
−8
−6
−4
m
y
p
u
i
w
t
e
o
uu
d
ê
n
éé
kp
gh
â
a
lv
:ââ
tp
ââ