Mandarin

This is the same as the permCantonese file, but has some additional mixed models analysis for comparing with the permutation test.

## permutation script 2019  
require(ggplot2)

## Loading required package: ggplot2

require(lme4)

## Loading required package: lme4

## Loading required package: Matrix

require(stringr)

## Loading required package: stringr

require(nlme) ## for lme()

## Loading required package: nlme

##   
## Attaching package: 'nlme'

## The following object is masked from 'package:lme4':  
##   
## lmList

require(lsmeans) ## for lsmeans()

## Loading required package: lsmeans

## Loading required package: emmeans

## The 'lsmeans' package is now basically a front end for 'emmeans'.  
## Users are encouraged to switch the rest of the way.  
## See help('transition') for more information, including how to  
## convert old 'lsmeans' objects and scripts to work with 'emmeans'.

# may need to run this command if the following doesn work install.packages("multcomp")  
require(multcomp) ## for multiple comparison stuff

## Loading required package: multcomp

## Loading required package: mvtnorm

## Loading required package: survival

## Loading required package: TH.data

## Loading required package: MASS

##   
## Attaching package: 'TH.data'

## The following object is masked from 'package:MASS':  
##   
## geyser

##   
## Attaching package: 'multcomp'

## The following object is masked from 'package:emmeans':  
##   
## cld

require(remef)

## Loading required package: remef

#Set your working directory  
# Go to Session in menu, go to Set Working Directory,   
# click on To Source File Location  
print(sessionInfo())

## R version 3.5.2 (2018-12-20)  
## Platform: x86\_64-apple-darwin15.6.0 (64-bit)  
## Running under: macOS Mojave 10.14.5  
##   
## Matrix products: default  
## BLAS: /Library/Frameworks/R.framework/Versions/3.5/Resources/lib/libRblas.0.dylib  
## LAPACK: /Library/Frameworks/R.framework/Versions/3.5/Resources/lib/libRlapack.dylib  
##   
## locale:  
## [1] en\_US.UTF-8/en\_US.UTF-8/en\_US.UTF-8/C/en\_US.UTF-8/en\_US.UTF-8  
##   
## attached base packages:  
## [1] stats graphics grDevices utils datasets methods base   
##   
## other attached packages:  
## [1] remef\_1.0.6.9000 multcomp\_1.4-8 TH.data\_1.0-9 MASS\_7.3-51.1   
## [5] survival\_2.43-3 mvtnorm\_1.0-8 lsmeans\_2.30-0 emmeans\_1.3.2   
## [9] nlme\_3.1-137 stringr\_1.4.0 lme4\_1.1-18-1 Matrix\_1.2-15   
## [13] ggplot2\_3.1.0   
##   
## loaded via a namespace (and not attached):  
## [1] Rcpp\_1.0.0 nloptr\_1.2.1 pillar\_1.3.1 compiler\_3.5.2   
## [5] plyr\_1.8.4 tools\_3.5.2 digest\_0.6.18 evaluate\_0.12   
## [9] tibble\_2.0.1 gtable\_0.2.0 lattice\_0.20-38 pkgconfig\_2.0.2   
## [13] rlang\_0.3.1 yaml\_2.2.0 xfun\_0.4 coda\_0.19-2   
## [17] withr\_2.1.2 dplyr\_0.8.0.1 knitr\_1.21 grid\_3.5.2   
## [21] tidyselect\_0.2.5 glue\_1.3.0 R6\_2.3.0 rmarkdown\_1.11   
## [25] minqa\_1.2.4 purrr\_0.2.5 magrittr\_1.5 codetools\_0.2-16  
## [29] scales\_1.0.0 htmltools\_0.3.6 splines\_3.5.2 assertthat\_0.2.0  
## [33] xtable\_1.8-3 colorspace\_1.4-0 sandwich\_2.5-0 estimability\_1.3  
## [37] stringi\_1.2.4 lazyeval\_0.2.1 munsell\_0.5.0 crayon\_1.3.4   
## [41] zoo\_1.8-4

apaformat <- function(p){  
 p = p + theme\_bw() # make the theme black-and-white rather than grey (do this before font changes, or it overrides them)  
 p = p + theme(  
# plot.title = theme\_text(face="bold", size=14), # use theme\_get() to see available options  
# axis.title.x = theme\_text(face="bold", size=12),  
# axis.title.y = theme\_text(face="bold", size=12, angle=90),  
 panel.grid.major = element\_blank(), # switch off major gridlines  
 panel.grid.minor = element\_blank(), # switch off minor gridlines  
# legend.position = c(0.2,0.8), # manually position the legend (numbers being from 0,0 at bottom left of whole plot to 1,1 at top right)  
# legend.title = theme\_blank(), # switch off the legend title  
# legend.text = theme\_text(size=12),  
# legend.key.size = unit(1.5, "lines"),  
 legend.key = element\_blank() # switch off the rectangle around symbols in the legend  
 )  
 return(p)  
}  
  
# data frame with onset for each region  
onsetStimCL = data.frame(SentenceType=c(rep("Subject",4),rep("Object",5)),cat = c("VN","DE","DCL","Head","N","V","DE","DCL","Head"), onset=c(0,0.808,1.02,1.49,0,0.431,0.767,1.027,1.427))  
onsetStimCL$HeadNoun = "DCL"  
onsetStimDE = data.frame(SentenceType=c(rep("Subject",3),rep("Object",4)),cat = c("VN","DE","Head","N","V","DE","Head"), onset=c(0,0.797,0.972,0,0.493,0.821,0.963))  
onsetStimDE$HeadNoun = "DE"  
onsetStim = rbind(onsetStimCL,onsetStimDE)  
onsetStim$time = onsetStim$onset \* 1000  
#onsetStim$time2=c(onsetStim$time[2:length(onsetStim$time)],0)  
#onsetStim$time2[onsetStim$cat=="Head"] = 1000\*c(2.001,1.876,1.479,1.501)  
onsetStim$time2 = c(805,1017,1487,2001, #subject DCL offsets  
 428,764,1024,1424,1876, # object DCL  
 794,969,1479, # subject DE  
 490, 818,960,1501) # object DE  
onsetStim$ypos = 0.95  
onsetStim$ypos[onsetStim$SentenceType == "Object"] = 0.75  
onsetStim$target = 1  
onsetStim$restarg = 0  
onsetStim[onsetStim$cat=="Head",]

## SentenceType cat onset HeadNoun time time2 ypos target restarg  
## 4 Subject Head 1.490 DCL 1490 2001 0.95 1 0  
## 9 Object Head 1.427 DCL 1427 1876 0.75 1 0  
## 12 Subject Head 0.972 DE 972 1479 0.95 1 0  
## 16 Object Head 0.963 DE 963 1501 0.75 1 0

aggregate(time2 ~ HeadNoun + cat, onsetStim, mean)

## HeadNoun cat time2  
## 1 DCL DCL 1455.5  
## 2 DCL DE 1020.5  
## 3 DE DE 964.5  
## 4 DCL Head 1938.5  
## 5 DE Head 1490.0  
## 6 DCL N 428.0  
## 7 DE N 490.0  
## 8 DCL V 764.0  
## 9 DE V 818.0  
## 10 DCL VN 805.0  
## 11 DE VN 794.0

bothdataAll=read.csv("ManRC eyetracking data-4yrs-final.csv")  
bothdata=bothdataAll  
xtabs(~ Participant,bothdata)

## Participant  
## 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18   
## 732 671 610 488 671 793 732 671 671 305 793 549 549 549 732 732 488 610   
## 19 20 21 22 23   
## 793 366 549 793 671

bothdata$HeadNoun <- factor(bothdata$HeadNoun,labels=c("DCL","DE"))  
bothdata$SentenceType=factor(bothdata$SentenceType,labels=c("Subject","Object"))  
  
bothdata$Item = str\_replace(bothdata$Item,"(S|O)","I")  
bothdata$time = as.numeric(as.character(bothdata$time))  
bothdata$time = bothdata$time \* 1000  
bothdata$target = bothdata$T  
#/(both$T+both$D)  
xtabs(~ Participant + HeadNoun , bothdata)

## HeadNoun  
## Participant DCL DE  
## 1 366 366  
## 2 366 305  
## 3 305 305  
## 4 244 244  
## 5 366 305  
## 6 427 366  
## 7 305 427  
## 8 244 427  
## 9 366 305  
## 10 0 305  
## 11 366 427  
## 12 305 244  
## 13 244 305  
## 14 305 244  
## 15 305 427  
## 16 427 305  
## 17 244 244  
## 18 305 305  
## 19 427 366  
## 20 366 0  
## 21 305 244  
## 22 488 305  
## 23 305 366

xtabs(~ Participant + SentenceType , bothdata)

## SentenceType  
## Participant Subject Object  
## 1 427 305  
## 2 427 244  
## 3 183 427  
## 4 305 183  
## 5 244 427  
## 6 305 488  
## 7 305 427  
## 8 366 305  
## 9 366 305  
## 10 183 122  
## 11 427 366  
## 12 488 61  
## 13 366 183  
## 14 305 244  
## 15 488 244  
## 16 427 305  
## 17 427 61  
## 18 183 427  
## 19 427 366  
## 20 183 183  
## 21 305 244  
## 22 427 366  
## 23 305 366

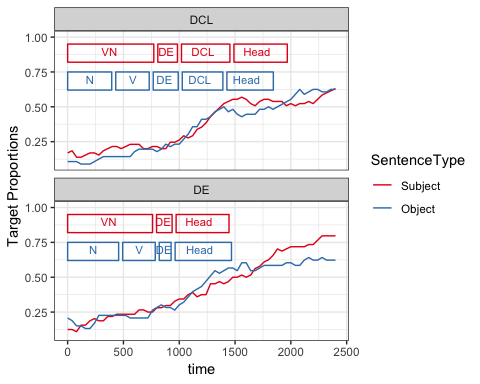
xtabs(~ Item + HeadNoun , bothdata)

## HeadNoun  
## Item DCL DE  
## I1 1464 0  
## I2 2074 0  
## I3 1891 0  
## I4 1952 0  
## I5 0 1647  
## I6 0 1952  
## I7 0 1647  
## I8 0 1891

xtabs(~ Item + SentenceType, bothdata)

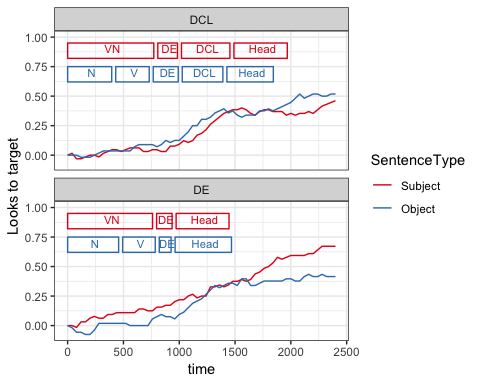
## SentenceType  
## Item Subject Object  
## I1 732 732  
## I2 915 1159  
## I3 1159 732  
## I4 1159 793  
## I5 854 793  
## I6 1037 915  
## I7 915 732  
## I8 1098 793

means.df = aggregate(cbind(target) ~ time + HeadNoun + SentenceType, bothdata, mean)  
p = ggplot(means.df , aes( x = time, y = target, colour=SentenceType))  
p = p + geom\_line()  
p = p + scale\_colour\_brewer(palette="Set1")  
p = p + theme\_bw() # change background to white  
p = p + ylab("Target Proportions")  
#p = p + ylim(0,1)  
p = p + facet\_wrap(~ HeadNoun, ncol=1)  
p = p + geom\_rect(data=onsetStim,aes(xmin=time,xmax = time2-33, ymin=ypos-0.13, ymax=ypos,colour=SentenceType),fill=NA, show.legend=FALSE)  
p = p + geom\_text(data=onsetStim,aes(x=(time+time2)/2, y=ypos-0.05, label=cat),hjust=0.7,size=3,show.legend=FALSE)  
p



Above is the raw data, below we remove the mean of time 0.

meansubset = subset(bothdata,time == 0)  
meanpartitemdf = aggregate(target ~ Participant + Item + HeadNoun + SentenceType, meansubset, mean)  
both = merge(bothdata,meanpartitemdf, all.x=TRUE,by=c("Participant","Item","HeadNoun","SentenceType"),sort=F)  
both$target = both$target.x-both$target.y  
# plot figure for dataset  
means.df = aggregate(cbind(target) ~ time + HeadNoun + SentenceType, both, mean)  
p = ggplot(means.df , aes( x = time, y = target, colour=SentenceType))  
p = p + geom\_line()  
p = p + scale\_colour\_brewer(palette="Set1")  
p = p + theme\_bw() # change background to white  
p = p + ylab("Looks to target")  
#p = p + ylim(0,1)  
p = p + facet\_wrap(~ HeadNoun, ncol=1)  
p = p + geom\_rect(data=onsetStim,aes(xmin=time,xmax = time2-33, ymin=ypos-0.13, ymax=ypos,colour=SentenceType),fill=NA, show.legend=FALSE)  
p = p + geom\_text(data=onsetStim,aes(x=(time+time2)/2, y=ypos-0.05, label=cat),size=3,show.legend=FALSE)  
p



#ggsave("new.png")

Here is a mixed model analysis using 200 ms windows

winsize = 200  
  
#unique(both$time)  
both$win = as.integer((both$time+10)/winsize) # like floor  
xtabs(~ win, both)

## win  
## 0 1 2 3 4 5 6 7 8 9 10 11 12   
## 1190 1190 1190 1190 1190 1190 1190 1190 1190 1190 1190 1190 238

both = subset(both, win != 12 & !is.na(time)) # not enough data in 12  
xtabs(~ win, both)

## win  
## 0 1 2 3 4 5 6 7 8 9 10 11   
## 1190 1190 1190 1190 1190 1190 1190 1190 1190 1190 1190 1190

xtabs(~ Item + SentenceType+ win, both)

## , , win = 0  
##   
## SentenceType  
## Item Subject Object  
## I1 60 60  
## I2 75 95  
## I3 95 60  
## I4 95 65  
## I5 70 65  
## I6 85 75  
## I7 75 60  
## I8 90 65  
##   
## , , win = 1  
##   
## SentenceType  
## Item Subject Object  
## I1 60 60  
## I2 75 95  
## I3 95 60  
## I4 95 65  
## I5 70 65  
## I6 85 75  
## I7 75 60  
## I8 90 65  
##   
## , , win = 2  
##   
## SentenceType  
## Item Subject Object  
## I1 60 60  
## I2 75 95  
## I3 95 60  
## I4 95 65  
## I5 70 65  
## I6 85 75  
## I7 75 60  
## I8 90 65  
##   
## , , win = 3  
##   
## SentenceType  
## Item Subject Object  
## I1 60 60  
## I2 75 95  
## I3 95 60  
## I4 95 65  
## I5 70 65  
## I6 85 75  
## I7 75 60  
## I8 90 65  
##   
## , , win = 4  
##   
## SentenceType  
## Item Subject Object  
## I1 60 60  
## I2 75 95  
## I3 95 60  
## I4 95 65  
## I5 70 65  
## I6 85 75  
## I7 75 60  
## I8 90 65  
##   
## , , win = 5  
##   
## SentenceType  
## Item Subject Object  
## I1 60 60  
## I2 75 95  
## I3 95 60  
## I4 95 65  
## I5 70 65  
## I6 85 75  
## I7 75 60  
## I8 90 65  
##   
## , , win = 6  
##   
## SentenceType  
## Item Subject Object  
## I1 60 60  
## I2 75 95  
## I3 95 60  
## I4 95 65  
## I5 70 65  
## I6 85 75  
## I7 75 60  
## I8 90 65  
##   
## , , win = 7  
##   
## SentenceType  
## Item Subject Object  
## I1 60 60  
## I2 75 95  
## I3 95 60  
## I4 95 65  
## I5 70 65  
## I6 85 75  
## I7 75 60  
## I8 90 65  
##   
## , , win = 8  
##   
## SentenceType  
## Item Subject Object  
## I1 60 60  
## I2 75 95  
## I3 95 60  
## I4 95 65  
## I5 70 65  
## I6 85 75  
## I7 75 60  
## I8 90 65  
##   
## , , win = 9  
##   
## SentenceType  
## Item Subject Object  
## I1 60 60  
## I2 75 95  
## I3 95 60  
## I4 95 65  
## I5 70 65  
## I6 85 75  
## I7 75 60  
## I8 90 65  
##   
## , , win = 10  
##   
## SentenceType  
## Item Subject Object  
## I1 60 60  
## I2 75 95  
## I3 95 60  
## I4 95 65  
## I5 70 65  
## I6 85 75  
## I7 75 60  
## I8 90 65  
##   
## , , win = 11  
##   
## SentenceType  
## Item Subject Object  
## I1 60 60  
## I2 75 95  
## I3 95 60  
## I4 95 65  
## I5 70 65  
## I6 85 75  
## I7 75 60  
## I8 90 65

xtabs(~ Participant + SentenceType, both)

## SentenceType  
## Participant Subject Object  
## 1 420 300  
## 2 420 240  
## 3 180 420  
## 4 300 180  
## 5 240 420  
## 6 300 480  
## 7 300 420  
## 8 360 300  
## 9 360 300  
## 10 180 120  
## 11 420 360  
## 12 480 60  
## 13 360 180  
## 14 300 240  
## 15 480 240  
## 16 420 300  
## 17 420 60  
## 18 180 420  
## 19 420 360  
## 20 180 180  
## 21 300 240  
## 22 420 360  
## 23 300 360

subjmeans.df = aggregate(target ~ win + HeadNoun + SentenceType + Participant + Item, both, mean)  
subjmeans.df$time=subjmeans.df$win\*winsize  
subjmeans.df$cwin = subjmeans.df$win - mean(subjmeans.df$win)  
subjmeans.df$cCL = ifelse(subjmeans.df$HeadNoun == "DCL",0.5,-0.5)  
subjmeans.df$cobject = ifelse(subjmeans.df$SentenceType == "Object",0.5,-0.5)  
subjmeans.df$Participant=factor(subjmeans.df$Participant)  
subjmeans.df$Item=factor(subjmeans.df$Item)  
mixmodel = lmer(target ~ cwin\*cCL\*cobject + (1 + cCL + cobject| Participant) + (1 | Item), subjmeans.df)  
summixed=summary(mixmodel)  
print(summixed)

## Linear mixed model fit by REML ['lmerMod']  
## Formula:   
## target ~ cwin \* cCL \* cobject + (1 + cCL + cobject | Participant) +   
## (1 | Item)  
## Data: subjmeans.df  
##   
## REML criterion at convergence: 3843.7  
##   
## Scaled residuals:   
## Min 1Q Median 3Q Max   
## -3.6031 -0.6039 -0.1224 0.7877 2.7321   
##   
## Random effects:  
## Groups Name Variance Std.Dev. Corr   
## Participant (Intercept) 0.010466 0.10231   
## cCL 0.034640 0.18612 -0.09   
## cobject 0.029798 0.17262 0.06 -0.38  
## Item (Intercept) 0.002002 0.04474   
## Residual 0.211719 0.46013   
## Number of obs: 2856, groups: Participant, 23; Item, 8  
##   
## Fixed effects:  
## Estimate Std. Error t value  
## (Intercept) 0.225531 0.028480 7.919  
## cwin 0.051160 0.002504 20.435  
## cCL -0.035798 0.053872 -0.664  
## cobject -0.049329 0.040766 -1.210  
## cwin:cCL -0.006487 0.005007 -1.296  
## cwin:cobject -0.001632 0.005007 -0.326  
## cCL:cobject 0.158270 0.037875 4.179  
## cwin:cCL:cobject 0.017542 0.010014 1.752  
##   
## Correlation of Fixed Effects:  
## (Intr) cwin cCL cobjct cwn:CL cwn:cb cCL:cb  
## cwin 0.000   
## cCL -0.050 0.000   
## cobject 0.066 0.000 -0.248   
## cwin:cCL 0.000 -0.018 0.000 0.000   
## cwin:cobjct 0.000 0.084 0.000 0.000 -0.011   
## cCL:cobject -0.020 0.000 0.039 -0.004 0.000 0.000   
## cwn:cCL:cbj 0.000 -0.011 0.000 0.000 0.084 -0.018 0.000

mixmodel2 = update(mixmodel, . ~ . - cwin:cCL:cobject)  
anova(mixmodel,mixmodel2)

## refitting model(s) with ML (instead of REML)

## Data: subjmeans.df  
## Models:  
## mixmodel2: target ~ cwin + cCL + cobject + (1 + cCL + cobject | Participant) +   
## mixmodel2: (1 | Item) + cwin:cCL + cwin:cobject + cCL:cobject  
## mixmodel: target ~ cwin \* cCL \* cobject + (1 + cCL + cobject | Participant) +   
## mixmodel: (1 | Item)  
## Df AIC BIC logLik deviance Chisq Chi Df Pr(>Chisq)   
## mixmodel2 15 3823.0 3912.3 -1896.5 3793.0   
## mixmodel 16 3821.9 3917.2 -1894.9 3789.9 3.0721 1 0.07965 .  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

Since we have a significant three way interaction of HeadNoun, SentenceType, and window, we can do posthocs. We use lsmeans to do contrasts between obj and sub for each window in each HeadNoun. lsmeans also adjusts p automatically for the number of comparisons. This is the “right” analysis if you wanted to use a traditional mix model. But it assumes that each window is independent and that is not right. Permutation test yields similar results without these incorrect assumptions.

options(contrasts = c("contr.treatment", "contr.poly"))  
# may need to run this command if the following doesn work install.packages("nlme")  
subjmeans.df$win = factor(subjmeans.df$win)  
subjmeans.df$HeadNoun = factor(subjmeans.df$HeadNoun)  
subjmeans.df$SentenceType = factor(subjmeans.df$SentenceType)  
mixmodelFactor = lmer(target ~ win\*HeadNoun\*SentenceType + (1 + HeadNoun+SentenceType| Participant) + (1+SentenceType | Item), subjmeans.df)  
#print(summary(mixmodelFactor))  
#save(mixmodelFactor,"mixmodelSentSlopeFactor.RData")  
  
model.lsmobj <- lsmeans(mixmodelFactor, ~ SentenceType | HeadNoun\* win)  
posthocs = summary(as.glht(pairs(model.lsmobj)))

## Note: df set to 31

print(posthocs)

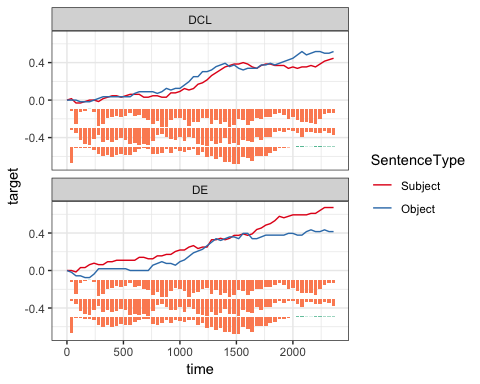
## $`HeadNoun = DCL, win = 0`  
##   
## Simultaneous Tests for General Linear Hypotheses  
##   
## Linear Hypotheses:  
## Estimate Std. Error t value Pr(>|t|)  
## Subject - Object == 0 0.01616 0.11831 0.137 0.892  
## (Adjusted p values reported -- single-step method)  
##   
##   
## $`HeadNoun = DE, win = 0`  
##   
## Simultaneous Tests for General Linear Hypotheses  
##   
## Linear Hypotheses:  
## Estimate Std. Error t value Pr(>|t|)  
## Subject - Object == 0 0.08913 0.11930 0.747 0.461  
## (Adjusted p values reported -- single-step method)  
##   
##   
## $`HeadNoun = DCL, win = 1`  
##   
## Simultaneous Tests for General Linear Hypotheses  
##   
## Linear Hypotheses:  
## Estimate Std. Error t value Pr(>|t|)  
## Subject - Object == 0 0.01319 0.11831 0.111 0.912  
## (Adjusted p values reported -- single-step method)  
##   
##   
## $`HeadNoun = DE, win = 1`  
##   
## Simultaneous Tests for General Linear Hypotheses  
##   
## Linear Hypotheses:  
## Estimate Std. Error t value Pr(>|t|)  
## Subject - Object == 0 0.1214 0.1193 1.018 0.317  
## (Adjusted p values reported -- single-step method)  
##   
##   
## $`HeadNoun = DCL, win = 2`  
##   
## Simultaneous Tests for General Linear Hypotheses  
##   
## Linear Hypotheses:  
## Estimate Std. Error t value Pr(>|t|)  
## Subject - Object == 0 0.03176 0.11831 0.268 0.79  
## (Adjusted p values reported -- single-step method)  
##   
##   
## $`HeadNoun = DE, win = 2`  
##   
## Simultaneous Tests for General Linear Hypotheses  
##   
## Linear Hypotheses:  
## Estimate Std. Error t value Pr(>|t|)  
## Subject - Object == 0 0.1294 0.1193 1.085 0.286  
## (Adjusted p values reported -- single-step method)  
##   
##   
## $`HeadNoun = DCL, win = 3`  
##   
## Simultaneous Tests for General Linear Hypotheses  
##   
## Linear Hypotheses:  
## Estimate Std. Error t value Pr(>|t|)  
## Subject - Object == 0 -0.01824 0.11831 -0.154 0.878  
## (Adjusted p values reported -- single-step method)  
##   
##   
## $`HeadNoun = DE, win = 3`  
##   
## Simultaneous Tests for General Linear Hypotheses  
##   
## Linear Hypotheses:  
## Estimate Std. Error t value Pr(>|t|)  
## Subject - Object == 0 0.1551 0.1193 1.3 0.203  
## (Adjusted p values reported -- single-step method)  
##   
##   
## $`HeadNoun = DCL, win = 4`  
##   
## Simultaneous Tests for General Linear Hypotheses  
##   
## Linear Hypotheses:  
## Estimate Std. Error t value Pr(>|t|)  
## Subject - Object == 0 -0.02994 0.11831 -0.253 0.802  
## (Adjusted p values reported -- single-step method)  
##   
##   
## $`HeadNoun = DE, win = 4`  
##   
## Simultaneous Tests for General Linear Hypotheses  
##   
## Linear Hypotheses:  
## Estimate Std. Error t value Pr(>|t|)  
## Subject - Object == 0 0.1347 0.1193 1.129 0.268  
## (Adjusted p values reported -- single-step method)  
##   
##   
## $`HeadNoun = DCL, win = 5`  
##   
## Simultaneous Tests for General Linear Hypotheses  
##   
## Linear Hypotheses:  
## Estimate Std. Error t value Pr(>|t|)  
## Subject - Object == 0 -0.05203 0.11831 -0.44 0.663  
## (Adjusted p values reported -- single-step method)  
##   
##   
## $`HeadNoun = DE, win = 5`  
##   
## Simultaneous Tests for General Linear Hypotheses  
##   
## Linear Hypotheses:  
## Estimate Std. Error t value Pr(>|t|)  
## Subject - Object == 0 0.1248 0.1193 1.046 0.304  
## (Adjusted p values reported -- single-step method)  
##   
##   
## $`HeadNoun = DCL, win = 6`  
##   
## Simultaneous Tests for General Linear Hypotheses  
##   
## Linear Hypotheses:  
## Estimate Std. Error t value Pr(>|t|)  
## Subject - Object == 0 -0.05544 0.11831 -0.469 0.643  
## (Adjusted p values reported -- single-step method)  
##   
##   
## $`HeadNoun = DE, win = 6`  
##   
## Simultaneous Tests for General Linear Hypotheses  
##   
## Linear Hypotheses:  
## Estimate Std. Error t value Pr(>|t|)  
## Subject - Object == 0 0.04768 0.11930 0.4 0.692  
## (Adjusted p values reported -- single-step method)  
##   
##   
## $`HeadNoun = DCL, win = 7`  
##   
## Simultaneous Tests for General Linear Hypotheses  
##   
## Linear Hypotheses:  
## Estimate Std. Error t value Pr(>|t|)  
## Subject - Object == 0 0.04264 0.11831 0.36 0.721  
## (Adjusted p values reported -- single-step method)  
##   
##   
## $`HeadNoun = DE, win = 7`  
##   
## Simultaneous Tests for General Linear Hypotheses  
##   
## Linear Hypotheses:  
## Estimate Std. Error t value Pr(>|t|)  
## Subject - Object == 0 0.04226 0.11930 0.354 0.726  
## (Adjusted p values reported -- single-step method)  
##   
##   
## $`HeadNoun = DCL, win = 8`  
##   
## Simultaneous Tests for General Linear Hypotheses  
##   
## Linear Hypotheses:  
## Estimate Std. Error t value Pr(>|t|)  
## Subject - Object == 0 0.0339 0.1183 0.287 0.776  
## (Adjusted p values reported -- single-step method)  
##   
##   
## $`HeadNoun = DE, win = 8`  
##   
## Simultaneous Tests for General Linear Hypotheses  
##   
## Linear Hypotheses:  
## Estimate Std. Error t value Pr(>|t|)  
## Subject - Object == 0 0.1041 0.1193 0.873 0.39  
## (Adjusted p values reported -- single-step method)  
##   
##   
## $`HeadNoun = DCL, win = 9`  
##   
## Simultaneous Tests for General Linear Hypotheses  
##   
## Linear Hypotheses:  
## Estimate Std. Error t value Pr(>|t|)  
## Subject - Object == 0 -0.01252 0.11831 -0.106 0.916  
## (Adjusted p values reported -- single-step method)  
##   
##   
## $`HeadNoun = DE, win = 9`  
##   
## Simultaneous Tests for General Linear Hypotheses  
##   
## Linear Hypotheses:  
## Estimate Std. Error t value Pr(>|t|)   
## Subject - Object == 0 0.2071 0.1193 1.736 0.0925 .  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
## (Adjusted p values reported -- single-step method)  
##   
##   
## $`HeadNoun = DCL, win = 10`  
##   
## Simultaneous Tests for General Linear Hypotheses  
##   
## Linear Hypotheses:  
## Estimate Std. Error t value Pr(>|t|)  
## Subject - Object == 0 -0.1105 0.1183 -0.934 0.357  
## (Adjusted p values reported -- single-step method)  
##   
##   
## $`HeadNoun = DE, win = 10`  
##   
## Simultaneous Tests for General Linear Hypotheses  
##   
## Linear Hypotheses:  
## Estimate Std. Error t value Pr(>|t|)   
## Subject - Object == 0 0.2351 0.1193 1.971 0.0577 .  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
## (Adjusted p values reported -- single-step method)  
##   
##   
## $`HeadNoun = DCL, win = 11`  
##   
## Simultaneous Tests for General Linear Hypotheses  
##   
## Linear Hypotheses:  
## Estimate Std. Error t value Pr(>|t|)  
## Subject - Object == 0 -0.08324 0.11831 -0.704 0.487  
## (Adjusted p values reported -- single-step method)  
##   
##   
## $`HeadNoun = DE, win = 11`  
##   
## Simultaneous Tests for General Linear Hypotheses  
##   
## Linear Hypotheses:  
## Estimate Std. Error t value Pr(>|t|)   
## Subject - Object == 0 0.2725 0.1193 2.284 0.0294 \*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
## (Adjusted p values reported -- single-step method)

subjmeans.df$pred <- remef(mixmodelFactor, ran = "all")  
means2.df = aggregate(cbind(pred,target) ~ time + HeadNoun + SentenceType, subjmeans.df, mean)  
timelist = as.integer(unique(subjmeans.df$time))  
p = ggplot(means2.df , aes( x = time, y = target, colour=SentenceType))  
meanwin= data.frame(time= rep(timelist,2),HeadNoun=rep(c("DCL","DE"),each=length(timelist)))  
#meanwin$time = (meanwin$win-1)\*winsize  
meanwin$pval = as.numeric(lapply(posthocs, function(x){ return( x$test$pvalues[1]) }) )  
meanwin$target = 0  
meanwin$pred =0  
meanwin$SentenceType = "Object"  
meansigPost = subset(meanwin,pval < 0.05)   
# color them grey  
p = p + geom\_rect(data=meansigPost,aes(xmin=time, xmax=time+winsize, ymin = 0, ymax= 1.0),fill="grey90",show.legend=FALSE)  
p = p + geom\_line()  
p = p + facet\_wrap(~ HeadNoun, ncol=1)  
p = p + scale\_colour\_brewer(palette="Set1")  
p = p + scale\_fill\_brewer(palette="Set2")  
p = p + geom\_vline(xintercept = seq(0,2400,by=winsize),colour="black", linetype = 2)  
#apaformat(p)

This is the permutation analysis

# create copy of data frame  
pdata = both[!is.na(both$time),]  
wdivsize = 1  
pdata$cobject = ifelse(pdata$SentenceType == "Subject",0.5,-0.5)  
pdata$cDErc = ifelse(pdata$HeadNoun == "DE",0.5,-0.5)  
pdata$win = as.integer(pdata$time/wdivsize)  
#xtabs(~ win, pdata)  
pdata$time = pdata$win\*wdivsize  
  
#pdata$target = pdata$restarg  
# create data frame which averages over subjects.   
# This also stores the results of the permutation analysis  
means.df = aggregate(target ~ SentenceType + cobject + time + HeadNoun, pdata, mean)  
means.df$pstr = 1000  
  
# We do this for each 100 ms window in the data  
timelist = unique(pdata$time)  
for (t in timelist){  
 # create data frame for ONE timebin for each HeadNounuage  
 onetime = subset(pdata,time == t)  
   
 # do regression model on target using structure SentenceTypeition  
 onemodel = summary(lm(target ~ cobject\*cDErc, onetime))  
 # print(summary(onemodel))  
 coefonemodel = coef(onemodel)  
 # this is the t-value for structure  
 objT = coefonemodel[2,3] # observed t-value  
 objP = abs(coefonemodel[2,4]) # observed p-value  
 means.df$objT[means.df$time == t] = objT  
 means.df$objP[means.df$time == t] = objP  
 deT = coefonemodel[3,3] # observed t-value  
 deP = abs(coefonemodel[3,4]) # observed p-value  
 means.df$deT[means.df$time == t] = deT  
 means.df$deP[means.df$time == t] = deP  
 intT = coefonemodel[4,3] # observed t-value  
 intP = abs(coefonemodel[4,4]) # observed p-value  
 means.df$intT[means.df$time == t] = intT  
 means.df$intP[means.df$time == t] = intP  
   
 randDCL = subset(onetime, HeadNoun=="DCL")  
 onemodelDCL = summary(lm(target ~ cobject, randDCL))  
 coefonemodelDCL = coef(onemodelDCL)  
 dclstrT = coefonemodelDCL[2,3] # observed t-value  
 randDE = subset(onetime, HeadNoun=="DE")  
 onemodelDE = summary(lm(target ~ cobject, randDE))  
 coefonemodelDE = coef(onemodelDE)  
 destrT = coefonemodelDE[2,3] # observed t-value  
 means.df$dclstrT[means.df$time == t] = dclstrT  
 means.df$destrT[means.df$time == t] = destrT  
}  
# to see these p-values, we draw them arbitrarily on the graph at 0.2.  
# when the p-value < 0.05, we draw a blue line above 0.2  
# when the p-value > 0.05, we draw an orange line below 0.2  
pliney = -0.1  
plinemax = 0.2  
means.df$plineobjP = pliney+plinemax\*(0.05-means.df$objP)  
means.df$plinecolobjP = ifelse(means.df$objP < 0.05,"a","b")  
plineydeP = -0.3  
means.df$plinedeP = plineydeP+plinemax\*(0.05-means.df$deP)  
means.df$plinecoldeP = ifelse(means.df$deP < 0.05,"a","b")  
plineyInt = -0.5  
means.df$plineintP = plineyInt+plinemax\*(0.05-means.df$intP)  
means.df$plinecolintP = ifelse(means.df$intP < 0.05,"a","b")  
  
  
wsize = mean(pdata$time[2:5] - pdata$time[1:4] - 8)/2  
means.df$SentenceType=factor(means.df$SentenceType,levels=c("Subject","Object"))  
p = ggplot(means.df , aes( x = time, y = target, colour=SentenceType))  
p = p + facet\_wrap(~ HeadNoun, ncol=1)  
p = p + geom\_line()  
p = p + scale\_colour\_brewer(palette="Set1")  
p = p + scale\_fill\_brewer(palette="Set2")  
p = p + theme\_bw() # change background to white  
p = p + geom\_rect(aes(xmin=time-wsize, xmax=time+wsize, ymin = pliney, ymax= plineobjP, fill=plinecolobjP),colour=NA,show.legend=FALSE)  
p = p + geom\_rect(aes(xmin=time-wsize, xmax=time+wsize, ymin = plineydeP, ymax= plinedeP, fill=plinecoldeP),colour=NA,show.legend=FALSE)  
p = p + geom\_rect(aes(xmin=time-wsize, xmax=time+wsize, ymin = plineyInt, ymax= plineintP, fill=plinecolintP),colour=NA,show.legend=FALSE)  
p

## Warning: Removed 4 rows containing missing values (geom\_rect).  
  
## Warning: Removed 4 rows containing missing values (geom\_rect).  
  
## Warning: Removed 4 rows containing missing values (geom\_rect).

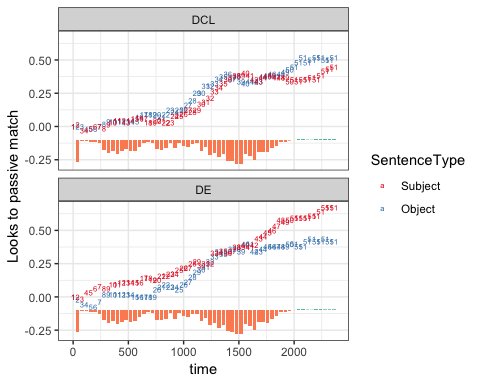


# Also, each window is not independent, so we create clusters for adjacent windows with p<0.05  
# cnum is the cluster number and we increment the number when the p value is > 0.05  
# so clusters with the same cnum are part of the same cluster  
cnum = 1  
lastpval = 100  
lasttdir = 1  
means.df$cnum = 1  
for (t in timelist[2:length(timelist)]){  
 onetime = subset(means.df,time == t & HeadNoun == "DCL" & SentenceType == "Object")  
 pval = abs(onetime$intP)  
 tdir = onetime$intT  
 if (pval < 0.05 & lastpval > 0.05 ){  
 cnum = cnum + 1 # increase cluster number when entering a significant cluster from a non-significant cluster  
 }  
 if (pval > 0.05 ){   
 cnum = cnum + 1 # increase cluster number when not significant  
 }else{  
 # if t value flips direction, even if both are signif,   
 # we should treat those as separate clusters  
 if (lasttdir\*tdir < 0){  
 cnum = cnum + 1   
 }  
 }  
 lastpval = pval  
 lasttdir = tdir  
 means.df$cnum[means.df$time == t] = cnum  
}  
head(means.df,10)

## SentenceType cobject time HeadNoun target pstr objT objP  
## 1 Object -0.5 0 DCL 0.00000000 1000 NaN NaN  
## 2 Subject 0.5 0 DCL 0.00000000 1000 NaN NaN  
## 3 Object -0.5 40 DCL 0.00000000 1000 1.4359235 0.1523596  
## 4 Subject 0.5 40 DCL 0.01538462 1000 1.4359235 0.1523596  
## 5 Object -0.5 80 DCL 0.00000000 1000 0.2501846 0.8026641  
## 6 Subject 0.5 80 DCL -0.03076923 1000 0.2501846 0.8026641  
## 7 Object -0.5 120 DCL -0.01785714 1000 1.2883987 0.1988797  
## 8 Subject 0.5 120 DCL -0.03076923 1000 1.2883987 0.1988797  
## 9 Object -0.5 160 DCL -0.01785714 1000 1.6248256 0.1055459  
## 10 Subject 0.5 160 DCL -0.01538462 1000 1.6248256 0.1055459  
## deT deP intT intP dclstrT destrT  
## 1 NaN NaN NaN NaN NaN NaN  
## 2 NaN NaN NaN NaN NaN NaN  
## 3 -1.4359235 0.1523596 0.1460261 0.88402649 0.92765144 1.099877  
## 4 -1.4359235 0.1523596 0.1460261 0.88402649 0.92765144 1.099877  
## 5 -1.0159650 0.3106954 1.7581830 0.08002373 -1.32226814 1.211492  
## 6 -1.0159650 0.3106954 1.7581830 0.08002373 -1.32226814 1.211492  
## 7 0.4001029 0.6894457 1.7323684 0.08452564 -0.34827214 1.949953  
## 8 0.4001029 0.6894457 1.7323684 0.08452564 -0.34827214 1.949953  
## 9 -0.1633831 0.8703579 1.5512424 0.12219500 0.05241213 2.230050  
## 10 -0.1633831 0.8703579 1.5512424 0.12219500 0.05241213 2.230050  
## plineobjP plinecolobjP plinedeP plinecoldeP plineintP plinecolintP  
## 1 NaN <NA> NaN <NA> NaN <NA>  
## 2 NaN <NA> NaN <NA> NaN <NA>  
## 3 -0.1204719 b -0.3204719 b -0.6668053 b  
## 4 -0.1204719 b -0.3204719 b -0.6668053 b  
## 5 -0.2505328 b -0.3521391 b -0.5060047 b  
## 6 -0.2505328 b -0.3521391 b -0.5060047 b  
## 7 -0.1297759 b -0.4278891 b -0.5069051 b  
## 8 -0.1297759 b -0.4278891 b -0.5069051 b  
## 9 -0.1111092 b -0.4640716 b -0.5144390 b  
## 10 -0.1111092 b -0.4640716 b -0.5144390 b  
## cnum  
## 1 1  
## 2 1  
## 3 2  
## 4 2  
## 5 3  
## 6 3  
## 7 4  
## 8 4  
## 9 5  
## 10 5

plineyInt=-0.1  
means.df$plineintP = plineyInt+plinemax\*(0.05-means.df$intP)  
# this shows the clusters  
p = ggplot(means.df , aes( x = time, y = target, colour=SentenceType, label=cnum))  
p = p + scale\_colour\_brewer(palette="Set1")  
p = p + scale\_fill\_brewer(palette="Set2")  
p = p + theme\_bw() # change background to white  
p = p + ylab("Looks to passive match")  
#p = p + ylim(0,1)  
p = p + facet\_wrap(~ HeadNoun, ncol=1)  
p = p + geom\_rect(aes(xmin=time-wsize, xmax=time+wsize, ymin = plineyInt, ymax= plineintP, fill=plinecolintP),colour=NA,show.legend=FALSE)  
p + geom\_text(size=2)

## Warning: Removed 4 rows containing missing values (geom\_rect).



# we now want to identify the clusters that were significant  
# p-values are same for active and passive, so we just used the active items.  
meansonlyact.df = subset(means.df, HeadNoun == "DCL" & SentenceType == "Object")  
sigcluster = subset(meansonlyact.df, abs(intP) < 0.05 )  
print(sigcluster,digits=3)

## SentenceType cobject time HeadNoun target pstr objT objP deT deP  
## 101 Object -0.5 2000 DCL 0.446 1000 0.713 0.476 1.289 0.199  
## 103 Object -0.5 2040 DCL 0.482 1000 0.489 0.625 1.012 0.313  
## 105 Object -0.5 2080 DCL 0.518 1000 0.351 0.726 0.667 0.505  
## 107 Object -0.5 2120 DCL 0.482 1000 0.337 0.736 1.157 0.249  
## 109 Object -0.5 2160 DCL 0.500 1000 0.294 0.769 1.148 0.252  
## 111 Object -0.5 2200 DCL 0.518 1000 0.200 0.841 1.011 0.313  
## 113 Object -0.5 2240 DCL 0.518 1000 0.610 0.542 1.014 0.312  
## 115 Object -0.5 2280 DCL 0.500 1000 1.014 0.311 1.260 0.209  
## 117 Object -0.5 2320 DCL 0.500 1000 1.241 0.216 1.033 0.302  
## 119 Object -0.5 2360 DCL 0.518 1000 1.224 0.222 0.813 0.417  
## intT intP dclstrT destrT plineobjP plinecolobjP plinedeP plinecoldeP  
## 101 1.97 0.0499 -0.914 1.85 -0.185 b -0.330 b  
## 103 2.42 0.0162 -1.422 1.98 -0.215 b -0.353 b  
## 105 2.55 0.0113 -1.617 1.98 -0.235 b -0.391 b  
## 107 2.05 0.0411 -1.265 1.62 -0.237 b -0.340 b  
## 109 2.02 0.0446 -1.251 1.60 -0.244 b -0.340 b  
## 111 2.37 0.0186 -1.575 1.77 -0.258 b -0.353 b  
## 113 2.37 0.0185 -1.271 2.07 -0.198 b -0.352 b  
## 115 2.13 0.0339 -0.803 2.20 -0.152 b -0.332 b  
## 117 2.16 0.0320 -0.672 2.32 -0.133 b -0.350 b  
## 119 2.17 0.0309 -0.696 2.32 -0.134 b -0.373 b  
## plineintP plinecolintP cnum  
## 101 -0.1000 a 51  
## 103 -0.0932 a 51  
## 105 -0.0923 a 51  
## 107 -0.0982 a 51  
## 109 -0.0989 a 51  
## 111 -0.0937 a 51  
## 113 -0.0937 a 51  
## 115 -0.0968 a 51  
## 117 -0.0964 a 51  
## 119 -0.0962 a 51

# this computes the sum of the t-values for each cluster  
sumcluster = aggregate(cbind(intT,dclstrT,destrT) ~ cnum + HeadNoun, meansonlyact.df, sum)  
head(sumcluster)

## cnum HeadNoun intT dclstrT destrT  
## 1 2 DCL 0.1460261 9.276514e-01 1.099877  
## 2 3 DCL 1.7581830 -1.322268e+00 1.211492  
## 3 4 DCL 1.7323684 -3.482721e-01 1.949953  
## 4 5 DCL 1.5512424 5.241213e-02 2.230050  
## 5 6 DCL 1.5619284 3.564950e-01 2.354545  
## 6 7 DCL 1.3220215 -4.370218e-17 1.754808

# here are the start and finish bits  
timedf = aggregate(time ~ cnum + HeadNoun,sigcluster,min)  
colnames(timedf)<-c("cnum","HeadNoun","starttime")  
timedf2 = aggregate(time ~ cnum + HeadNoun,sigcluster,max)  
timedf$endtime = timedf2$time+33  
print(timedf)

## cnum HeadNoun starttime endtime  
## 1 51 DCL 2000 2393

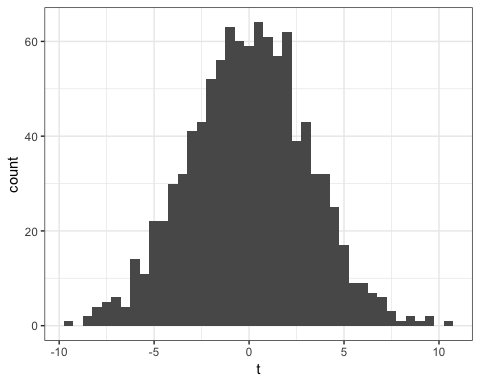
paste(timedf$starttime,"-",timedf$endtime,"ms",sep="",collapse=",")

## [1] "2000-2393ms"

# now we create a distribution of t-values (save in permdist)  
# by randomly scrambling the active and passive labels for each time window 1000 times  
createPermDist <- function(filename="permDist.RData"){  
 n = 1000  
 exptests = data.frame()  
 for (s in 1:length(sigcluster$time)){  
 # print(cl)  
 cl = sigcluster$cnum[s] # cluster number  
 b = sigcluster$time[s] # time  
 print(paste("b ",b))  
 # one time point  
 onetime = subset(pdata, time %in% b)  
 # randSet is a copy of onetime that is scrambled  
 randSet = onetime  
   
 for (i in 1:n){  
 # set.seed(i)  
 # randomly scramble cobject labels without replacement  
 randSet$cobject = sample(randSet$cobject,length(randSet$cobject))  
 randSet$cDErc = sample(randSet$cDErc,length(randSet$cDErc))  
 # test if target is related to random scrambled cobject  
 onemodel = summary(lm(target ~ cobject\*cDErc, randSet))  
 coefonemodel = coef(onemodel)  
 intT = as.numeric(coefonemodel[4,3]) # observed t-value  
 randDCL = subset(randSet, HeadNoun=="DCL")  
 onemodelDCL = summary(lm(target ~ cobject, randDCL))  
 coefonemodelDCL = coef(onemodelDCL)  
 dclstrT = as.numeric(coefonemodelDCL[2,3]) # observed t-value  
 randDE = subset(randSet, HeadNoun=="DE")  
 onemodelDE = summary(lm(target ~ cobject, randDE))  
 coefonemodelDE = coef(onemodelDE)  
 destrT = as.numeric(coefonemodelDE[2,3]) # observed t-value  
   
 df = data.frame(t=intT,cluster=cl,time=b,sim=i,dclT=dclstrT,deT=destrT)  
 exptests = rbind(exptests, df )  
 }  
 }  
 save(exptests,file=filename)  
 return(exptests)  
}  
#exptests = createPermDist("permDistMan5.RData") # since this takes a lot of time, we save the values in a file.  
load("permDistMan5.RData") #this creates data frame exptests from file  
  
# we sum over clusters so that longer clusters have stronger t-values  
sumt.df = aggregate(cbind(t,dclT,deT) ~cluster + sim, exptests, sum)  
head(sumt.df)

## cluster sim t dclT deT  
## 1 51 1 -0.6514369 -5.1960576 1.056916  
## 2 51 2 0.3968350 -0.7606289 2.382795  
## 3 51 3 -4.8081372 0.2780874 -2.904489  
## 4 51 4 -3.7834474 -1.8828483 2.835628  
## 5 51 5 -1.0611246 -2.1225112 5.470188  
## 6 51 6 -3.2347268 2.6891421 -2.965886

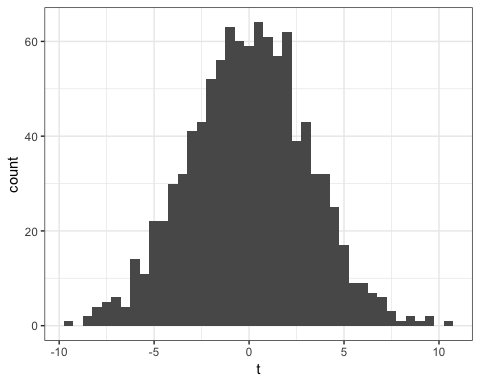
# simulated sum cluster histogram  
p = ggplot(sumt.df,aes(x = t))  
p = p +geom\_histogram(binwidth=0.5)  
p+theme\_bw()



# this code extracts out the maximum sum t for each simulation at each age  
if (length(unique(sumt.df$cluster)) > 1){ # only run if there is more than one cluster  
maxclusterdist = data.frame()  
 for (s in unique(sumt.df$sim)) {  
 # get all results for one simulation in one HeadNounuage  
 onesim = subset(sumt.df,sim == s)  
 onesim$absT = abs(onesim$t)  
 onesim$absdclT = abs(as.numeric(onesim$dclT))  
 onesim$absdeT = abs(as.numeric(onesim$deT))  
 # find max t-value  
 maxrow = onesim[order(onesim$absT,decreasing = T),]  
 maxclusterdist = rbind(maxclusterdist,maxrow[1,])  
 }  
}else{  
 maxclusterdist = sumt.df  
}  
head(maxclusterdist)

## cluster sim t dclT deT  
## 1 51 1 -0.6514369 -5.1960576 1.056916  
## 2 51 2 0.3968350 -0.7606289 2.382795  
## 3 51 3 -4.8081372 0.2780874 -2.904489  
## 4 51 4 -3.7834474 -1.8828483 2.835628  
## 5 51 5 -1.0611246 -2.1225112 5.470188  
## 6 51 6 -3.2347268 2.6891421 -2.965886

# Shows the simulated distribution with maximum cluster t values  
maxclusterdist2 = maxclusterdist[order(maxclusterdist$t),]  
#end = data.frame(xint = maxclusterdist2[c(25,975,1025,1975),]$t)  
p = ggplot(maxclusterdist,aes(x = t))  
p = p +geom\_histogram(binwidth=0.5)  
#p = p +geom\_vline(end,mapping=aes(xintercept=xint))  
p+theme\_bw()

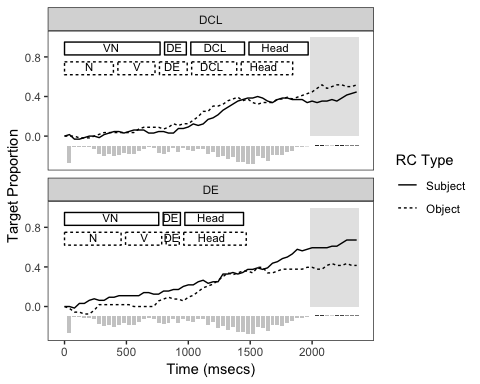


# maxclusterdist is sorted by HeadNounuage,   
# this identifies tvalues that are greater than dist t-values  
for (cl in unique(sumcluster$cnum)){  
 bins = unique(means.df[means.df$cnum == cl,]$time)  
  
 # permtdist is the dist t-values,   
 # so p value is proportion of values greater than observed t-value.  
 # absolute value gives two sided test  
 intT = abs(sumcluster$intT[sumcluster$cnum==cl])  
 intP = sum(abs(maxclusterdist$t) > intT, na.rm = TRUE)/length(maxclusterdist$t)  
 if (intP < 0.05){  
 print(paste("Cluster Interaction Term ",cl,"Obs.sum t",round(intT,3),"PropDist > observed p-value",intP))  
 }  
   
 destrT = abs(sumcluster$destrT[sumcluster$cnum==cl])  
 deT = sum(abs(maxclusterdist$deT) > destrT, na.rm = TRUE)/length(maxclusterdist$deT)  
 if (deT < 0.05){  
 print(paste("Separate Test of deT only",cl,"Obs.sum t",round(destrT,3),"PropDist > observed p-value",deT))  
 }  
   
 dclstrT = abs(sumcluster$dclstrT[sumcluster$cnum==cl])  
 dclT = sum(abs(maxclusterdist$dclT) > dclstrT, na.rm = TRUE)/length(maxclusterdist$dclT)  
 if (dclT < 0.05){  
 print(paste("Separate Test of dclT only",cl,"Obs.sum t",round(dclstrT,3),"PropDist > observed p-value",dclT))  
 }  
  
 means.df$permtestp[means.df$time %in% bins] = intP  
}

## [1] "Cluster Interaction Term 51 Obs.sum t 22.224 PropDist > observed p-value 0"  
## [1] "Separate Test of deT only 51 Obs.sum t 19.694 PropDist > observed p-value 0"  
## [1] "Separate Test of dclT only 51 Obs.sum t 11.484 PropDist > observed p-value 0"

# now we update our plot  
p = ggplot(means.df , aes( x = time, y = target, linetype=SentenceType))  
p = p + facet\_wrap(~ HeadNoun, ncol=1)  
# this pulls out the clusters which are significant by the permutation test  
meansigStr = subset(means.df,permtestp < 0.025)   
# color them grey  
if (length(meansigStr$time) > 0){  
 p = p + geom\_rect(data=meansigStr,aes(xmin=time-wsize-4, xmax=time+wsize+4, ymin = pliney+0.1, ymax= 1.0),colour=NA,fill="grey90",show.legend=FALSE)  
}  
# same as before  
p = p + geom\_line()  
#p = p + scale\_colour\_brewer(palette="Set1")  
#p = p + scale\_fill\_brewer(palette="Set2")  
p = p + scale\_linetype\_discrete(name="RC Type")  
p = p + theme\_bw() # change background to white  
p = p + ylab("Target Proportion")  
p = p + xlab("Time (msecs)")  
#p = p + ylim(0,1)  
p = p + geom\_rect(aes(xmin=time-wsize, xmax=time+wsize, ymin = plineyInt, ymax= plineintP, fill=plinecolintP),colour=NA,show.legend=FALSE)  
#p = p + geom\_curve(data=meansigPost,aes(x=time+10, xend=time+winsize-10, y = 0.9, yend= 0.9),color="black",size=1, lineend = "square", curvature = -0.5, show.legend=FALSE)  
p = p + geom\_rect(data=onsetStim,aes(xmin=time,xmax = time2-33, ymin=ypos-0.13, ymax=ypos,linetype=SentenceType),fill=NA,colour="black", show.legend=FALSE)  
p = p + geom\_text(data=onsetStim,aes(x=(time+time2)/2, y=ypos-0.05, label=cat), hjust=0.7, size=3,show.legend=FALSE)  
p = p +scale\_colour\_grey()  
p = p +scale\_fill\_grey()  
apaformat(p)

## Warning: Removed 4 rows containing missing values (geom\_rect).



ggsave("permCan.png",width=8,height=6)

## Warning: Removed 4 rows containing missing values (geom\_rect).

Here is some code that will report the results.

modelComparison <- function(model,modellist=list(),verbose=0,filename="modellist.RData"){  
 # print(model)  
 terms = attr(terms(model),"term.labels")  
 modellist = append(modellist,model)  
 # print(terms)  
 if (length(terms) > 0){  
 newformula = paste(". ~ . - ",terms[length(terms)],"")  
 print(paste("remove",newformula))  
 model2 = update(model, as.formula(newformula))  
 if (verbose > 0){  
 print(summary(model2))  
 }  
 am1=anova(model2, model)  
 print(am1)  
 sig = ifelse(am1$`Pr(>Chisq)`[2]<0.05," \*\*\* ","")  
 print(paste("########## Above comparison for ",terms[length(terms)],sig))  
   
 terms = attr(terms(model),"term.labels")  
 modellist = modelComparison(model2,modellist)  
 }  
 save(modellist, file= filename)  
 return(modellist)  
}  
  
getModelNumHighestTerm <- function(varname,modellist) {  
 for (i in 1:length(modellist)) {  
 terms = attr(terms(modellist[[i]]), "term.labels")  
   
 if (length(terms) > 0 & terms[length(terms)] == varname) {  
 return(i)  
 }  
 }  
 return(-1)  
}  
  
report <- function(varname, modellist){  
 fullmodel=modellist[[1]]  
 model.sum = summary(fullmodel)  
 model.coef = coefficients(model.sum)  
 outstr=""  
 if (varname %in% rownames(model.coef)){  
 oneline = model.coef[varname,]  
 outstr = paste("$\\beta$=",round(oneline[[1]],3),", SE=",round(oneline[[2]],4),sep="")  
 }  
   
 tm = getModelNumHighestTerm(varname,modellist)  
 model.anova = anova(modellist[[tm]],modellist[[tm+1]])  
 chisqr = round(model.anova$Chisq[2],2)  
 chisqrdf = model.anova$"Chi Df"[2]  
 chisqrp = model.anova$"Pr(>Chisq)"[2]  
   
 chisqrpf = format.pval(pv = chisqrp, digits = 2,eps=0.001,nsmall = 3)  
 if (substr(chisqrpf,1,1)!='<'){  
 chisqrpf = paste("=",chisqrpf,sep="")  
 }  
# outstr = paste(c("beta","SE","z","p"),oneline,collapse=", ",sep="=")  
 if (outstr != ""){  
 outstr=paste(outstr,", ",sep="")  
 }  
 outstr = paste(outstr,"$\\chi^2$(",chisqrdf,")=",chisqr,", p",chisqrpf,sep="")  
 return(outstr)   
}  
  
modellist = modelComparison(mixmodel) # create anova table

## [1] "remove . ~ . - cwin:cCL:cobject "

## refitting model(s) with ML (instead of REML)

## Data: subjmeans.df  
## Models:  
## model2: target ~ cwin + cCL + cobject + (1 + cCL + cobject | Participant) +   
## model2: (1 | Item) + cwin:cCL + cwin:cobject + cCL:cobject  
## model: target ~ cwin \* cCL \* cobject + (1 + cCL + cobject | Participant) +   
## model: (1 | Item)  
## Df AIC BIC logLik deviance Chisq Chi Df Pr(>Chisq)   
## model2 15 3823.0 3912.3 -1896.5 3793.0   
## model 16 3821.9 3917.2 -1894.9 3789.9 3.0721 1 0.07965 .  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
## [1] "########## Above comparison for cwin:cCL:cobject "  
## [1] "remove . ~ . - cCL:cobject "

## refitting model(s) with ML (instead of REML)

## Data: subjmeans.df  
## Models:  
## model2: target ~ cwin + cCL + cobject + (1 + cCL + cobject | Participant) +   
## model2: (1 | Item) + cwin:cCL + cwin:cobject  
## model: target ~ cwin + cCL + cobject + (1 + cCL + cobject | Participant) +   
## model: (1 | Item) + cwin:cCL + cwin:cobject + cCL:cobject  
## Df AIC BIC logLik deviance Chisq Chi Df Pr(>Chisq)   
## model2 14 3838.3 3921.7 -1905.2 3810.3   
## model 15 3823.0 3912.3 -1896.5 3793.0 17.382 1 3.057e-05 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
## [1] "########## Above comparison for cCL:cobject \*\*\* "  
## [1] "remove . ~ . - cwin:cobject "

## refitting model(s) with ML (instead of REML)

## Data: subjmeans.df  
## Models:  
## model2: target ~ cwin + cCL + cobject + (1 + cCL + cobject | Participant) +   
## model2: (1 | Item) + cwin:cCL  
## model: target ~ cwin + cCL + cobject + (1 + cCL + cobject | Participant) +   
## model: (1 | Item) + cwin:cCL + cwin:cobject  
## Df AIC BIC logLik deviance Chisq Chi Df Pr(>Chisq)  
## model2 13 3836.4 3913.9 -1905.2 3810.4   
## model 14 3838.3 3921.7 -1905.2 3810.3 0.0857 1 0.7697  
## [1] "########## Above comparison for cwin:cobject "  
## [1] "remove . ~ . - cwin:cCL "

## refitting model(s) with ML (instead of REML)

## Data: subjmeans.df  
## Models:  
## model2: target ~ cwin + cCL + cobject + (1 + cCL + cobject | Participant) +   
## model2: (1 | Item)  
## model: target ~ cwin + cCL + cobject + (1 + cCL + cobject | Participant) +   
## model: (1 | Item) + cwin:cCL  
## Df AIC BIC logLik deviance Chisq Chi Df Pr(>Chisq)  
## model2 12 3836.5 3908.0 -1906.3 3812.5   
## model 13 3836.4 3913.9 -1905.2 3810.4 2.0945 1 0.1478  
## [1] "########## Above comparison for cwin:cCL "  
## [1] "remove . ~ . - cobject "

## refitting model(s) with ML (instead of REML)

## Data: subjmeans.df  
## Models:  
## model2: target ~ cwin + cCL + (1 + cCL + cobject | Participant) + (1 |   
## model2: Item)  
## model: target ~ cwin + cCL + cobject + (1 + cCL + cobject | Participant) +   
## model: (1 | Item)  
## Df AIC BIC logLik deviance Chisq Chi Df Pr(>Chisq)  
## model2 11 3835.9 3901.4 -1907.0 3813.9   
## model 12 3836.5 3908.0 -1906.3 3812.5 1.3946 1 0.2376  
## [1] "########## Above comparison for cobject "  
## [1] "remove . ~ . - cCL "

## refitting model(s) with ML (instead of REML)

## Data: subjmeans.df  
## Models:  
## model2: target ~ cwin + (1 + cCL + cobject | Participant) + (1 | Item)  
## model: target ~ cwin + cCL + (1 + cCL + cobject | Participant) + (1 |   
## model: Item)  
## Df AIC BIC logLik deviance Chisq Chi Df Pr(>Chisq)  
## model2 10 3835.2 3894.8 -1907.6 3815.2   
## model 11 3835.9 3901.4 -1907.0 3813.9 1.2936 1 0.2554  
## [1] "########## Above comparison for cCL "  
## [1] "remove . ~ . - cwin "

## refitting model(s) with ML (instead of REML)

## Data: subjmeans.df  
## Models:  
## model2: target ~ (1 + cCL + cobject | Participant) + (1 | Item)  
## model: target ~ cwin + (1 + cCL + cobject | Participant) + (1 | Item)  
## Df AIC BIC logLik deviance Chisq Chi Df Pr(>Chisq)   
## model2 9 4223.4 4277.0 -2102.7 4205.4   
## model 10 3835.2 3894.8 -1907.6 3815.2 390.18 1 < 2.2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
## [1] "########## Above comparison for cwin \*\*\* "

# here are the start and finish bits  
timedf = aggregate(time ~ cnum + HeadNoun,meansigStr,min)  
colnames(timedf)<-c("cnum","HeadNoun","starttime")  
timedf2 = aggregate(time ~ cnum + HeadNoun,meansigStr,max)  
timedf$endtime = timedf2$time+33  
print(timedf)

## cnum HeadNoun starttime endtime  
## 1 51 DCL 2000 2393  
## 2 51 DE 2000 2393

# significant regions in posthoc analysis  
print(meansigPost)

## time HeadNoun pval target pred SentenceType  
## 24 2200 DE 0.02936692 0 0 Object

print(summixed)

## Linear mixed model fit by REML ['lmerMod']  
## Formula:   
## target ~ cwin \* cCL \* cobject + (1 + cCL + cobject | Participant) +   
## (1 | Item)  
## Data: subjmeans.df  
##   
## REML criterion at convergence: 3843.7  
##   
## Scaled residuals:   
## Min 1Q Median 3Q Max   
## -3.6031 -0.6039 -0.1224 0.7877 2.7321   
##   
## Random effects:  
## Groups Name Variance Std.Dev. Corr   
## Participant (Intercept) 0.010466 0.10231   
## cCL 0.034640 0.18612 -0.09   
## cobject 0.029798 0.17262 0.06 -0.38  
## Item (Intercept) 0.002002 0.04474   
## Residual 0.211719 0.46013   
## Number of obs: 2856, groups: Participant, 23; Item, 8  
##   
## Fixed effects:  
## Estimate Std. Error t value  
## (Intercept) 0.225531 0.028480 7.919  
## cwin 0.051160 0.002504 20.435  
## cCL -0.035798 0.053872 -0.664  
## cobject -0.049329 0.040766 -1.210  
## cwin:cCL -0.006487 0.005007 -1.296  
## cwin:cobject -0.001632 0.005007 -0.326  
## cCL:cobject 0.158270 0.037875 4.179  
## cwin:cCL:cobject 0.017542 0.010014 1.752  
##   
## Correlation of Fixed Effects:  
## (Intr) cwin cCL cobjct cwn:CL cwn:cb cCL:cb  
## cwin 0.000   
## cCL -0.050 0.000   
## cobject 0.066 0.000 -0.248   
## cwin:cCL 0.000 -0.018 0.000 0.000   
## cwin:cobjct 0.000 0.084 0.000 0.000 -0.011   
## cCL:cobject -0.020 0.000 0.039 -0.004 0.000 0.000   
## cwn:cCL:cbj 0.000 -0.011 0.000 0.000 0.084 -0.018 0.000

In this analysis, there was a main effect of window, =0.051, SE=0.0025, (1)=390.18, p<0.001, and a marginal three-way interaction of window, RC type, and Head Noun =0.018, SE=0.01, (1)=3.07, p=0.080. To explore this three-way interaction, posthoc comparisons were performed comparing subject and object RC types in each window in both studies (p-values were adjusted for the 24 multiple comparisons). The only significant differences between subject and object was in the 2200-2400 window in the DE condition.

accdf = read.csv("ManRC\_accuracy\_data\_all\_final.csv")  
  
accdf$cCL = ifelse(accdf$structure == "DemCL",0.5,-0.5)  
accdf$cobject = ifelse(accdf$extraction == "object",0.5,-0.5)  
accdf$participant=factor(accdf$participant)  
accdf$item = str\_replace(accdf$item,"(S|O)","I")  
accdf$item=factor(accdf$item)  
xtabs( ~ structure + participant,accdf)

## participant  
## structure 1 2 3 4 5 6 7 8 9 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25  
## bare 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 8 8  
## DemCL 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 0 0  
## participant  
## structure 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 44 45 46  
## bare 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8  
## DemCL 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

xtabs( ~ extraction + participant,accdf)

## participant  
## extraction 1 2 3 4 5 6 7 8 9 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25  
## object 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4  
## subject 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4  
## participant  
## extraction 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 44 45 46  
## object 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4  
## subject 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4

xtabs( ~ structure + item,accdf)

## item  
## structure I1 I2 I3 I4 I5 I6 I7 I8  
## bare 0 0 0 0 44 44 44 44  
## DemCL 44 44 44 44 0 0 0 0

xtabs( ~ extraction + item,accdf)

## item  
## extraction I1 I2 I3 I4 I5 I6 I7 I8  
## object 22 22 22 22 22 22 22 22  
## subject 22 22 22 22 22 22 22 22

acc.glm = glmer(Correct ~ cobject\*cCL + (1 + extraction | participant) + (1 | item),accdf,family="binomial")  
print(summary(acc.glm))

## Generalized linear mixed model fit by maximum likelihood (Laplace  
## Approximation) [glmerMod]  
## Family: binomial ( logit )  
## Formula: Correct ~ cobject \* cCL + (1 + extraction | participant) + (1 |   
## item)  
## Data: accdf  
##   
## AIC BIC logLik deviance df.resid   
## 450.8 481.7 -217.4 434.8 344   
##   
## Scaled residuals:   
## Min 1Q Median 3Q Max   
## -2.3237 -1.0366 0.5140 0.6778 1.0806   
##   
## Random effects:  
## Groups Name Variance Std.Dev. Corr   
## participant (Intercept) 0.61285 0.7828   
## extractionsubject 2.30679 1.5188 -1.00  
## item (Intercept) 0.01764 0.1328   
## Number of obs: 352, groups: participant, 44; item, 8  
##   
## Fixed effects:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 0.7868 0.1360 5.786 7.21e-09 \*\*\*  
## cobject -0.5128 0.3415 -1.501 0.133   
## cCL 0.1445 0.2616 0.552 0.581   
## cobject:cCL -0.2394 0.6664 -0.359 0.719   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Correlation of Fixed Effects:  
## (Intr) cobjct cCL   
## cobject -0.060   
## cCL 0.051 -0.024   
## cobject:cCL -0.022 0.027 -0.027

accdf$pred = remef(acc.glm, ran = "all")  
head(accdf)

## item participant extraction Correct structure cCL cobject pred  
## 1 I1 1 subject 0 DemCL 0.5 -0.5 -0.4123535  
## 2 I2 1 subject 1 DemCL 0.5 -0.5 1.9350090  
## 3 I3 1 subject 1 DemCL 0.5 -0.5 1.9657572  
## 4 I4 1 subject 1 DemCL 0.5 -0.5 1.9554244  
## 5 I1 1 object 1 DemCL 0.5 0.5 1.4737056  
## 6 I2 1 object 1 DemCL 0.5 0.5 1.4115142

meandf = aggregate(Correct ~ extraction + structure, accdf, mean)  
print(meandf)

## extraction structure Correct  
## 1 object bare 0.6136364  
## 2 subject bare 0.6931818  
## 3 object DemCL 0.6136364  
## 4 subject DemCL 0.7386364

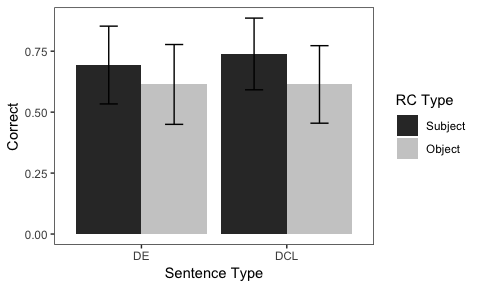
# compute sd from pred without random effects  
meandf$sd = aggregate(pred ~ extraction + structure, accdf,sd)$pred  
print(meandf)

## extraction structure Correct sd  
## 1 object bare 0.6136364 1.0861246  
## 2 subject bare 0.6931818 1.0582441  
## 3 object DemCL 0.6136364 1.0546698  
## 4 subject DemCL 0.7386364 0.9755494

# se is computed from sd divided by number of participants  
meandf$se = meandf$sd/sqrt( nlevels(accdf$participant) )  
meandf$upper = meandf$Correct + meandf$se  
meandf$lower = meandf$Correct - meandf$se  
print(meandf)

## extraction structure Correct sd se upper lower  
## 1 object bare 0.6136364 1.0861246 0.1637394 0.7773758 0.4498969  
## 2 subject bare 0.6931818 1.0582441 0.1595363 0.8527181 0.5336455  
## 3 object DemCL 0.6136364 1.0546698 0.1589975 0.7726338 0.4546389  
## 4 subject DemCL 0.7386364 0.9755494 0.1470696 0.8857060 0.5915668

meandf$extraction=factor(meandf$extraction,labels=c("Subject","Object"),levels=c("subject","object"))  
meandf$structure=factor(meandf$structure,labels=c("DE","DCL"))  
  
p = ggplot(meandf, aes(x=structure,y=Correct, fill=extraction,ymin = lower, ymax=upper))  
p = p +geom\_bar(stat="identity",position="dodge")  
p = p +scale\_fill\_grey(name="RC Type")  
p = p + xlab("Sentence Type")  
p = p + geom\_errorbar(width=0.25, position=position\_dodge(.9) )  
apaformat(p)



ggsave("accuracy.png",width=5,height=3)