22q_subcort_volumes

Charles Schleifer

Jan 11, 2024

Overview

Subcortical nuclei volumes for Thalamus, Hippocampus, and Amygdala were generated with FreeSurfer 7.3.2 segment_structures. This script uses Generalized Additive Mixed Models to analyze 22q11.2 CNV gene dosage effects and developmental trajectories.

Citation: Charles H. Schleifer, Kathleen P. O'Hora, Hoki Fung, Jennifer Xu, Taylor-Ann Robinson, Angela S. Wu, Leila Kushan-Wells, Amy Lin, Christopher R. K. Ching, Carrie E. Bearden. Effects of Gene Dosage and Development on Subcortical Nuclei Volumes in Individuals with 22q11.2 Copy Number Variations https://doi.org/10.1101/2023.10.31.564553

set up workspace

```
# clear workspace
rm(list = ls(all.names = TRUE))

# install longCombat for first time
#install.packages("devtools")
#devtools::install_github("jcbeer/longCombat")

# install limma
#install.packages("BiocManager")
#BiocManager::install("limma")

# install DMwR
#install.packages("remotes")
#remotes::install_github("cran/DMwR")

# list packages to load
packages <- c("mgcViz", "conflicted", "here", "magrittr", "dplyr", "stringr",
 ↪  "tidyr", "data.table", "readxl", "tableone", "ggplot2", "gghalves", "gt",
 ↪  "ggnewscale", "viridis", "patchwork", "DMwR", "lmerTest", "effects",
 ↪  "longCombat", "mgcv", "gratia", "tidymv", "formatR")

# install packages if not yet installed
all_packages <- rownames(installed.packages())
installed_packages <- packages %in% all_packages
if (any(installed_packages ==
 ↪  FALSE)){install.packages(packages[!installed_packages])}

# load packages
invisible(lapply(packages, library, character.only = TRUE))
```

```
# use the filter function from dplyr, not stats
conflict_prefer("filter", "dplyr")

# get path to project repo directory
project <- here()
print(paste("Project directory:", project))
```

read MRI data

```
# path to data (organized with subject directories containing only the relevant
 ↪  files)
dpath <- file.path(project, "volume_data")

# get files
sessions_all <- list.files(path=dpath, pattern= "Q_[0-9]")

read_fssubcort_long <- function(path, sesh, name){
  file <- file.path(dpath, sesh, name)
  orig <- read.table(file)
  out <- as.data.frame(t(orig$V2))
  names(out) <- orig$V1
  #out$MRI_S_ID <- sesh
  # for longitudinal need to remove end of folder name to get MRI_S_ID
  out$MRI_S_ID <- gsub(".long.*", "", sesh)
  return(out)
}

# read data
thal_lr <- lapply(sessions_all, function(s) read_fssubcort_long(path=dpath,
 ↪  sesh=s, name = "ThalamicNuclei.long.volumes.txt")) %>% do.call(rbind,.)
names(thal_lr)[which(!names(thal_lr)== "MRI_S_ID")] <- paste0("Thal_",
 ↪  names(thal_lr)[which(!names(thal_lr)== "MRI_S_ID")])
amy_l <- lapply(sessions_all, function(s) read_fssubcort_long(path=dpath, sesh=s,
 ↪  name = "lh.amygNucVolumes.long.txt")) %>% do.call(rbind,.)
names(amy_l)[which(!names(amy_l)== "MRI_S_ID")] <- paste0("Amy_Left_",
 ↪  names(amy_l)[which(!names(amy_l)== "MRI_S_ID")])
amy_r <- lapply(sessions_all, function(s) read_fssubcort_long(path=dpath, sesh=s,
 ↪  name = "rh.amygNucVolumes.long.txt")) %>% do.call(rbind,.)
names(amy_r)[which(!names(amy_r)== "MRI_S_ID")] <- paste0("Amy_Right_",
 ↪  names(amy_r)[which(!names(amy_r)== "MRI_S_ID")])
hip_l <- lapply(sessions_all, function(s) read_fssubcort_long(path=dpath, sesh=s,
 ↪  name = "lh.hippoSfVolumes.long.txt")) %>% do.call(rbind,.)
names(hip_l)[which(!names(hip_l)== "MRI_S_ID")] <- paste0("Hip_Left_",
 ↪  names(hip_l)[which(!names(hip_l)== "MRI_S_ID")])
hip_r <- lapply(sessions_all, function(s) read_fssubcort_long(path=dpath, sesh=s,
 ↪  name = "rh.hippoSfVolumes.long.txt")) %>% do.call(rbind,.)
names(hip_r)[which(!names(hip_r)== "MRI_S_ID")] <- paste0("Hip_Right_",
 ↪  names(hip_r)[which(!names(hip_r)== "MRI_S_ID")])

# merge data
subcort_all <- merge(x =thal_lr, y = amy_l, by = "MRI_S_ID")
```

```
subcort_all <- merge(x =subcort_all, y = amy_r, by = "MRI_S_ID")
subcort_all <- merge(x =subcort_all, y =hip_l, by = "MRI_S_ID")
subcort_all <- merge(x =subcort_all, y =hip_r, by = "MRI_S_ID")

# replace problematic characters with underscores in column names
colnames(subcort_all) <- gsub("-", "_", colnames(subcort_all))
colnames(subcort_all) <- gsub("\\(", "_", colnames(subcort_all))
colnames(subcort_all) <- gsub("\\)", "_", colnames(subcort_all))

# read eTIV
etiv <- read.csv(file.path(dpath,"etiv_cross_sectional.txt"), header =TRUE)
```

exclude data and combine regions

```
# store original subcort_all
subcort_all_orig <- subcort_all

# manually exclude subjects based on QC
exclude <- c("Q_0071_03022010", "Q_0258_06062014", "Q_0288_03212017",
 ↪  "Q_0397_11082018", "Q_0549_10182022", "Q_0315_02162017", "Q_0167_04052012",
 ↪  "Q_0021_08172009", "Q_0361_08212017")
subcort_all <- subcort_all %>% filter(!MRI_S_ID %in% exclude)

# exclude all medial pulvinar data (poor segmentation in most images), and thal Pc
 ↪  and Pt (too small on average for good segmentation)
subcort_all <- subset(subcort_all, select =
 ↪  -c(Thal_Left_PuM,Thal_Right_PuM,Thal_Left_Pc,Thal_Right_Pc,Thal_Left_Pt,Thal_Right_Pt))

# exclude whole hippocampal body and head (redundant with more fine grained
 ↪  regions)
subcort_all <- subset(subcort_all, select = -c(Hip_Right_Whole_hippocampal_body,
 ↪  Hip_Left_Whole_hippocampal_body, Hip_Right_Whole_hippocampal_head,
 ↪  Hip_Left_Whole_hippocampal_head))

# exclude bilateral thalamus from n= 4 sessions that have visibly poor thalamic
 ↪  segmentation (part of striatum appears to be included in some lateral thalamic
 ↪  ROIs)
thal_exclude <- c("Q_0244_09162013", "Q_0244_09232014", "Q_0304_12202016",
 ↪  "Q_0425_11052019")
thal_names <- names(subcort_all)[grep("Thal", names(subcort_all))]
thal_ex_rows <- which(subcort_all$MRI_S_ID %in% thal_exclude)
subcort_all[thal_ex_rows, thal_names] <- NA

# group subregions
thal_combine <- list(MD_all = c("MDm", "MDl"),
                     VL_all = c("VLp", "VLa"),
                     VA_all = c("VA", "VAmc"),
                     Pu_part= c("PuA", "PuL", "PuI"))

hip_combine <- list(CA1= c("CA1_head", "CA1_body"),
                     CA3= c("CA3_head", "CA3_body"),
                     CA4= c("CA4_head", "CA4_body"),
                     GC_ML_DG= c("GC_ML_DG_head", "GC_ML_DG_body"),
                     molecular_layer_HP= c("molecular_layer_HP_head",
                      ↪  "molecular_layer_HP_body"),
```

```
                    presubiculum= c("presubiculum_head", "presubiculum_body"),
                    subiculum= c("subiculum_head", "subiculum_body"))

# make groupings for left and right
# L thal
thal_combine_L <-  thal_combine
names(thal_combine_L) <- paste0("Thal_Left_", names(thal_combine))
thal_combine_L <- lapply(thal_combine_L, function(x) paste0("Thal_Left_", x))
# R thal
thal_combine_R <-  thal_combine
names(thal_combine_R) <- paste0("Thal_Right_", names(thal_combine))
thal_combine_R <- lapply(thal_combine_R, function(x) paste0("Thal_Right_", x))
# L hip
hip_combine_L <-  hip_combine
names(hip_combine_L) <- paste0("Hip_Left_", names(hip_combine))
hip_combine_L <- lapply(hip_combine_L, function(x) paste0("Hip_Left_", x))
# R hip
hip_combine_R <-  hip_combine
names(hip_combine_R) <- paste0("Hip_Right_", names(hip_combine))
hip_combine_R <- lapply(hip_combine_R, function(x) paste0("Hip_Right_", x))

# list of all columns to replace
all_replace <- c(as.vector(unlist(hip_combine_R)),
 ↪  as.vector(unlist(hip_combine_L)), as.vector(unlist(thal_combine_R)),
 ↪  as.vector(unlist(thal_combine_L)))

# function to combine subregions
# regions should be vector of column names to add together
combine_volumes <- function(regions, df){
  dfcols <- df[, c(regions)]
  out <- data.frame(name = rowSums(dfcols))
  return(out)
}

# combine subregions
# L thal
combined_thal_L <- lapply(thal_combine_L, function(x)
 ↪  combine_volumes(df=subcort_all, regions = x))
df_combined_thal_L <- do.call(cbind, combined_thal_L)
colnames(df_combined_thal_L) <- names(combined_thal_L)
# R thal
combined_thal_R <- lapply(thal_combine_R, function(x)
 ↪  combine_volumes(df=subcort_all, regions = x))
df_combined_thal_R <- do.call(cbind, combined_thal_R)
colnames(df_combined_thal_R) <- names(combined_thal_R)
# L hip
combined_hip_L <- lapply(hip_combine_L, function(x)
 ↪  combine_volumes(df=subcort_all, regions = x))
df_combined_hip_L <- do.call(cbind, combined_hip_L)
colnames(df_combined_hip_L) <- names(combined_hip_L)
# R hip
combined_hip_R <- lapply(hip_combine_R, function(x)
 ↪  combine_volumes(df=subcort_all, regions = x))
```

```
df_combined_hip_R <- do.call(cbind, combined_hip_R)
colnames(df_combined_hip_R) <- names(combined_hip_R)

# remove unnecessary columns and add new columns to subcort all
subcort_all <- subcort_all[, which(!names(subcort_all) %in% all_replace)]

# add new columns
subcort_all <- cbind(subcort_all, df_combined_thal_L, df_combined_thal_R,
 ↪  df_combined_hip_L, df_combined_hip_R)

# get feature names
sc_names <- names(subcort_all)[which(names(subcort_all) != "MRI_S_ID")]
```

histograms for each region

```
#lapply(sc_names, function(n)hist(subcort_all[, n], main=n))
```

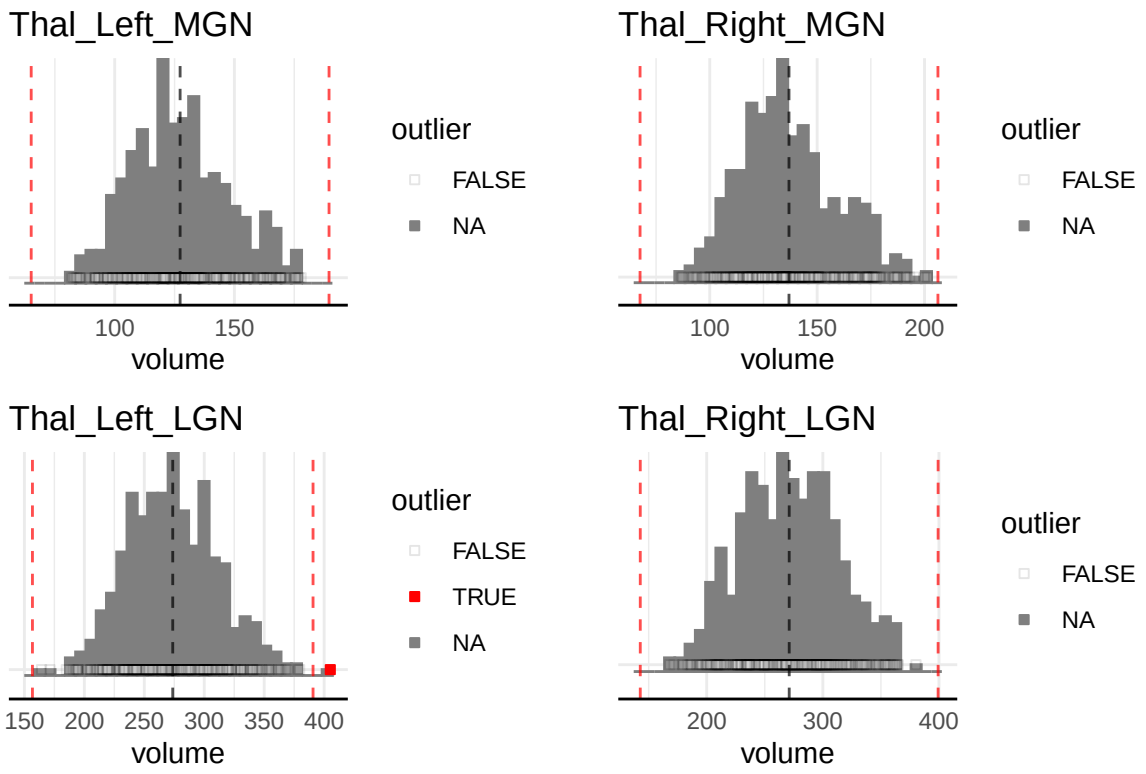
outlier from standard deviation

```
sc_long  <- reshape2::melt(subcort_all, id.vars = "MRI_S_ID", measure.vars
 ↪  =sc_names, value.name = "volume")

# get mean, sd, and upper and lower bounds (mean +/- 3*SD) for each roi
roi_mean_sd <- data.frame(roi=sc_names)
roi_mean_sd$mean <- lapply(sc_names, function(n) mean(subcort_all[, n],
 ↪  na.rm=TRUE)) %>% do.call(rbind,.)
roi_mean_sd$sd <- lapply(sc_names, function(n) sd(subcort_all[, n], na.rm=TRUE))
 ↪  %>% do.call(rbind,.)
roi_mean_sd$lower <- (roi_mean_sd$mean - (3*roi_mean_sd$sd))
roi_mean_sd$upper <- (roi_mean_sd$mean + (3*roi_mean_sd$sd))

# for each measurement, determine if outside bounds for roi
sc_out_long <- sc_long
for(i in 1:nrow(sc_out_long)){
  roi <- sc_out_long[i,"variable"]
  vol <- sc_out_long[i,"volume"]
  mean <- roi_mean_sd[which(roi_mean_sd$roi== roi),"mean"]
  upper <- roi_mean_sd[which(roi_mean_sd$roi== roi),"upper"]
  lower <- roi_mean_sd[which(roi_mean_sd$roi== roi),"lower"]
  sc_out_long[i,"roi_mean"] <- mean
  sc_out_long[i,"lower"] <- lower
  sc_out_long[i,"upper"] <- upper
  sc_out_long[i,"outlier"] <- (vol >= upper | vol <= lower)
}

# get outliers
out_filter <- filter(sc_out_long, outlier == TRUE)

# get non-outliers
sc_long_inlier <- filter(sc_out_long, outlier == FALSE)[, c("MRI_S_ID",
 ↪  "variable", "volume")]
```

```
outlier_plot <- function(roi){
  df <- filter(sc_out_long, variable == roi)
```

```
  ggplot(df, aes(x =volume, y = "", color =outlier, fill =outlier))+
  #geom_point(shape = 21, alpha = 0.8, position = position_jitter(width = 0,
   ↪  height= 0.01))+
  geom_histogram(inherit.aes = FALSE, aes(x =volume), fill = "grey50", color =
   ↪  "grey50")+
  #geom_jitter(shape = 21, alpha = 0.7, width = 0, height = 1)+
  geom_point(shape = 22, aes(alpha =outlier))+
  geom_vline(aes(xintercept = upper), lty = "dashed", color = "red", alpha = 0.7)+
  geom_vline(aes(xintercept = lower), lty = "dashed", color = "red", alpha = 0.7)+
  geom_vline(aes(xintercept = roi_mean), lty = "dashed", color = "black", alpha =
   ↪  0.7)+
  scale_color_manual(values = c("black", "red"))+
  scale_fill_manual(values = c("white", "red"))+
  scale_alpha_manual(values = c(0.1,1))+
  ggtitle(roi)+
  ylab("")+
  theme_minimal()+
  theme(axis.line.x = element_line())+
  scale_y_discrete(expand = c(5,0))
}

oplots <- lapply(sc_names, outlier_plot)
names(oplots) <- sc_names

oplots_out <- (oplots$Thal_Left_MGN + oplots$Thal_Right_MGN)/(oplots$Thal_Left_LGN
 ↪  + oplots$Thal_Right_LGN)+plot_annotation(title =element_text("Outlier plots
 ↪  for thalamus MGN and LGN"))
oplots_out
```

Outlier plots for thalamus MGN and LGN




```
#ggsave(oplots_out, file =file.path(project,
 ↪  "figures/review_response/outlier_plots.png"), device = "png", width= 7,
 ↪  height= 4)
```

merge hemispheres by averaging whenever bilateral data are available

```
# match left and right hemispheres in same row
sc_long_hemi <- sc_long_inlier
sc_long_hemi$bl_region <-  sc_long_hemi$variable %>% gsub("_Left", "",.) %>%
 ↪  gsub("_Right", "",.)
sc_long_hemi$hemi <-  str_extract(string=sc_long_hemi$variable, pattern=
 ↪  "Left|Right")
sc_long_hemi_merged <- merge(x =filter(sc_long_hemi, hemi== "Left"), y
 ↪  =filter(sc_long_hemi, hemi== "Right"), by = c("MRI_S_ID", "bl_region"), all.x
 ↪  =TRUE, all.y =TRUE)

# average left and right
sc_long_hemi_avg <- sc_long_hemi_merged
sc_long_hemi_avg$bl_volume <- NA
for(i in 1:nrow(sc_long_hemi_avg)){
  sc_long_hemi_avg[i,"bl_volume"] <- mean(x = c(sc_long_hemi_avg[i,"volume.x"],
 ↪  sc_long_hemi_avg[i,"volume.y"]), na.rm=TRUE)
}

# convert back to wide df for input to ComBat
setDT(sc_long_hemi_avg)
subcort_bilat <- dcast(sc_long_hemi_avg, MRI_S_ID ~ bl_region, value.var =
 ↪  "bl_volume")
```

lookup table for ROI info

```
# read edited lookup table matching freesurfer names to full names
#lut <- read.csv(file.path(project, "lut_names_thal_hip_amy_match.txt"), header
 ↪  =TRUE)
lut <- read.csv(file.path(project, "lut_names_thal_hip_amy_match_combine.csv"),
 ↪  header =TRUE)
lut$region_match <- gsub("-", "_", lut$region)
lut$region_match <- gsub("\\(", "_", lut$region_match)
lut$region_match <- gsub("\\)", "_", lut$region_match)

# edit lut for bilateral matching
lut_bilat <- lut
lut_bilat$bilat_match <- lut_bilat$region_match %>% gsub("Right_", "",.) %>%
 ↪  gsub("Left_", "",.)
lut_bilat$bilat_name <- lut_bilat$Name %>% gsub("right", "",.) %>% gsub("left",
 ↪  "",.)
lut_bilat_n <- lut_bilat[, c("bilat_name", "bilat_match", "Structure")]
lut_unique <- lut_bilat_n[!duplicated(lut_bilat_n),]
lut_unique
```

```
##                            bilat_name                  bilat_match
## 1                    lateral geniculate                          LGN
## 3                     medial geniculate                          MGN
## 5                     pulvinar inferior                          PuI
## 6                       pulvinar medial                          PuM
## 7             limitans (suprageniculate)                        L_Sg
## 8                 ventral posterolateral                          VPL
## 9                           centromedian                           CM
## 10              ventral lateral anterior                          VLa
## 11                    pulvinar anterior                          PuA
## 12                   mediodorsal medial                          MDm
## 13                       parafascicular                           Pf
## 14                  ventral anterior mc                        VAmc
## 15                  mediodorsal lateral                          MDl
## 16                       central medial                          CeM
## 17                     ventral anterior                           VA
## 18             medial ventral (reuniens)                      MV_Re_
## 19                          ventromedial                           VM
## 20                     central lateral                           CL
## 21                     pulvinar lateral                          PuL
## 22                           paratenial                           Pt
## 23                      anteroventral                           AV
## 24                        paracentral                           Pc
## 25             ventral lateral posterior                          VLp
## 26                    lateral posterior                           LP
## 49                         laterodorsal                           LD
## 51                       whole thalamus              Whole_thalamus
## 53                     hippocampal tail            Hippocampal_tail
## 54                       subiculum body              subiculum_body
## 55                            CA1 body                    CA1_body
## 56                       subiculum head              subiculum_head
## 57                    hippocampal fissure         hippocampal_fissure
## 58                      presubiculum head            presubiculum_head
```

```
## 59                            CA1 head                   CA1_head
## 60                   presubiculum body           presubiculum_body
## 61                       parasubiculum               parasubiculum
## 62                 molecular layer head     molecular_layer_HP_head
## 63                 molecular layer body     molecular_layer_HP_body
## 64                        GC ML DG head               GC_ML_DG_head
## 65                             CA3 body                    CA3_body
## 66                        GC ML DG body               GC_ML_DG_body
## 67                             CA4 head                    CA4_head
## 68                             CA4 body                    CA4_body
## 69                              fimbria                     fimbria
## 70                             CA3 head                    CA3_head
## 71 hippocampal amygdala transition area                        HATA
## 72                    whole hippocampus          Whole_hippocampus
## 73                whole hippocampus body      Whole_hippocampal_body
## 74                whole hippocampus head      Whole_hippocampal_head
## 75                       lateral nucleus             Lateral_nucleus
## 76                         basal nucleus               Basal_nucleus
## 77                accessory basal nucleus     Accessory_Basal_nucleus
## 78               anterior amygdaloid area Anterior_amygdaloid_area_AAA
## 79                       central nucleus             Central_nucleus
## 80                        medial nucleus              Medial_nucleus
## 81                      cortical nucleus            Cortical_nucleus
## 82           corticoamygdaloid transition  Corticoamygdaloid_transitio
## 83                    paralaminar nucleus         Paralaminar_nucleus
## 84                         whole amygdala              Whole_amygdala
## 85                            mediodorsal                      MD_all
## 86                        ventral lateral                      VL_all
## 87                        ventral anterior                     VA_all
## 88                                pulvinar                    Pu_part
## 93                                     CA1                         CA1
## 94                                  CA2/3                         CA3
## 95                                     CA4                         CA4
## 96                                GC ML DG                  GC_ML_DG
## 97                         molecular layer          molecular_layer_HP
## 98                            presubiculum                presubiculum
## 99                               subiculum                   subiculum
##      Structure
## 1     thalamus
## 3     thalamus
## 5     thalamus
## 6     thalamus
## 7     thalamus
## 8     thalamus
## 9     thalamus
## 10    thalamus
## 11    thalamus
## 12    thalamus
## 13    thalamus
## 14    thalamus
## 15    thalamus
## 16    thalamus
## 17    thalamus
## 18    thalamus
```

```
## 19    thalamus
## 20    thalamus
## 21    thalamus
## 22    thalamus
## 23    thalamus
## 24    thalamus
## 25    thalamus
## 26    thalamus
## 49    thalamus
## 51    thalamus
## 53 hippocampus
## 54 hippocampus
## 55 hippocampus
## 56 hippocampus
## 57 hippocampus
## 58 hippocampus
## 59 hippocampus
## 60 hippocampus
## 61 hippocampus
## 62 hippocampus
## 63 hippocampus
## 64 hippocampus
## 65 hippocampus
## 66 hippocampus
## 67 hippocampus
## 68 hippocampus
## 69 hippocampus
## 70 hippocampus
## 71 hippocampus
## 72 hippocampus
## 73 hippocampus
## 74 hippocampus
## 75    amygdala
## 76    amygdala
## 77    amygdala
## 78    amygdala
## 79    amygdala
## 80    amygdala
## 81    amygdala
## 82    amygdala
## 83    amygdala
## 84    amygdala
## 85    thalamus
## 86    thalamus
## 87    thalamus
## 88    thalamus
## 93 hippocampus
## 94 hippocampus
## 95 hippocampus
## 96 hippocampus
## 97 hippocampus
## 98 hippocampus
## 99 hippocampus
```

```
# add eTIVnormed
lut_unique <- rbind(lut_unique, data.frame(bilat_name = "total ICV", bilat_match=
 ↪  "eTIVnormed", Structure = "whole brain"))
```

load sistat data and get lists of scans to use

all sistat tables should be exported as CSVs into a single directory the next several chunks deal with reading, cleaning and annotating the data exported from sistat, and then age matching the hcs sample is younger than del due to a large amount of very young hcs subjects. plan is to match samples by using followup timepoints rather than baseline for some younger participants, and dropping several older del subjects, and younger hcs subjects (prioritizing dropping subjects with worse motion stats when possible)

```
# set location of directory with ucla sistat CSVs
csvdir_ucla <- file.path(project, "demographics/ucla_sistat")

# get list of files_ucla in directory
files_ucla <- list.files(csvdir_ucla)
fpaths <- lapply(files_ucla, function(file) paste(csvdir_ucla, file, sep= "/"))

# clean names
fnames <- gsub(".csv", "", files_ucla)
fnames <- gsub("Re22Q_", "", fnames)
fnames <- gsub("Form_", "", fnames)
fnames <- gsub("Qry_", "", fnames)

# read all, set to na: "-9999", "-9998","."
input_all_ucla <- lapply(fpaths, read.csv, header =T, na.strings =
 ↪  c(".","-9999","-9998"), strip.white =T, sep= ",")
names(input_all_ucla) <- fnames
df_all_ucla <- lapply(input_all_ucla, function(x) data.frame(x))

# get demo_mri
ucla_demo <- df_all_ucla$demo_mri

# remove "FAMILY MEMBER" designation from subject identity
ucla_demo$SUBJECT_IDENTITY <- ucla_demo$SUBJECT_IDENTITY %>% sub("FAMILY
 ↪  MEMBER","",.) %>% sub(",", "",.) %>% trimws(which= "both") %>% as.factor
# change sex coding from 0/1 to F/M and set to factor
ucla_demo$SEX <- factor(ucla_demo$SEX, levels = c(0,1), labels = c("F","M"))
```

read temporary csv with several subjects not yet in sistat

```
# TODO: this chunk is temporary until sistat is updated
# TODO: note: q_0526 sex was "na" in original sheet, changed to M because combat
 ↪  can't have NAs
# read new data
temp_demo <- read_xlsx(file.path(project,
 ↪  "demographics/temporary/sMRI_demo_info_forCharlie.xlsx"), col_names =TRUE, na
 ↪  = "", trim_ws = TRUE)

# make empty demographics data frame to add new data to
demo_add <- ucla_demo[1:nrow(temp_demo),]
demo_add[,] <- NA
demo_add$SUBJECTID <- temp_demo$`Subject ID`
```

```
demo_add$SUBJECT_IDENTITY <- temp_demo$Diagnosis
demo_add$MRI_S_ID <- temp_demo$`MRI ID`
demo_add$SEX <- as.factor(temp_demo$Sex)
demo_add$AGE <- temp_demo$Age
demo_add$AGEMONTH <- temp_demo$Age*12
demo_add$CONVERTEDVISITNUM <- 2

# append to ucla demo
ucla_demo <- rbind(ucla_demo, demo_add)
```

continue regular steps

```
# manually fix missing sex for Q_0381_09102019
# TODO: fix in sistat and re-export
ucla_demo[which(ucla_demo$MRI_S_ID == "Q_0381_09102019"),"SEX"] <- "F"

# set race =NA to 7 (unknown)
ucla_demo$RACE[is.na(ucla_demo$RACE)] <- 7
# set race as factor 1=American Indian/Alaska Native; 2=Asian; 3=Native
 ↪  Hawaiian/Pacific Islander; 4=Black or African American; 5=White; 6=Multiple;
 ↪  7=Unknown
ucla_demo$RACE <- factor(ucla_demo$RACE, levels = c(1:7), labels =
 ↪  c("1_Native_American", "2_Asian", "3_Pacific_Island", "4_Black", "5_White",
 ↪  "6_Multiple", "7_Unknown"))
# ethnicity as factor with 0=N 1=Y
ucla_demo$HISPANIC[is.na(ucla_demo$HISPANIC)] <- "Unknown"
ucla_demo$HISPANIC <- factor(ucla_demo$HISPANIC, levels = c(0,1,"Unknown"), labels
 ↪  = c("N", "Y", "Unknown"))
# get more accurate age with AGEMONTH/12
ucla_demo$AGE <- as.numeric(ucla_demo$AGEMONTH)/12

# subset to used scans
ucla_demo <- filter(ucla_demo, MRI_S_ID %in% subcort_all$MRI_S_ID)

# function to add column to code timepoints relative to sample used (i.e. if visit
 ↪  1 and 1.12 missing, then 1.24 is baseline)
# trio/prisma coded as T/P-visit_n where T-1 would be the subject's first trio
 ↪  scan and P-1 the first prisma, P-2 the second...
# function should be applied to the indicies of rows (r) in a subset of demo_mri
gettp <- function(r, df){
  sub <- df$SUBJECTID[[r]]
  visit <- df$CONVERTEDVISITNUM[[r]]
  all_visits <- df$CONVERTEDVISITNUM[which(df$SUBJECTID == sub)] %>% sort
  n_visits <- length(all_visits)
  nt_visits <-length(which(all_visits < 2))
  np_visits <- length(which(all_visits >= 2))
  visit_index <- which(all_visits == visit)
  if (visit < 2){
    label =paste("T-", visit_index, sep= "")
  }else if (visit >= 2){
    p_visits <- all_visits[which(all_visits >= 2)] %>% sort
    p_visit_index <- which(p_visits == visit)
    label =paste("P-", p_visit_index, sep= "")
  }
```

```
  return(c(sub, visit, label, n_visits, nt_visits, np_visits, visit_index))
}

# get timepoints
timepoints <- lapply(1:nrow(ucla_demo), function(r) gettp(r, ucla_demo)) %>%
 ↪  do.call(rbind,.) %>% as.data.frame
colnames(timepoints) <- c("SUBJECTID", "CONVERTEDVISITNUM", "converted_timepoint",
 ↪  "n_timepoints", "n_trio", "n_prisma", "visit_index")
ucla_demo_tp <- cbind(ucla_demo, timepoints[,3:7])
ucla_demo_tp$visit_index %<>% as.factor

# subset to under max age limit (45 years old)
ucla_demo_tp_agelim <- filter(ucla_demo_tp, ucla_demo_tp$AGE < 50)
#ucla_demo_tp_agelim <- filter(ucla_demo_tp, ucla_demo_tp$AGE < 44)

# subset to hcs del
#ucla_demo_hcs_del <- ucla_demo_tp_agelim %>% filter(SUBJECT_IDENTITY== "CONTROL"
 ↪  | SUBJECT_IDENTITY == "PATIENT-DEL")

# remove unused factor levels
ucla_demo_tp_agelim %<>% droplevels
```

All timepoints, demographics summary

```
demo_summary <- CreateTableOne(data =ucla_demo_tp_agelim, vars = c("AGE", "SEX"),
 ↪  strata = "SUBJECT_IDENTITY", addOverall =F)
print(demo_summary, showAllLevels =T)
```

```
##                   Stratified by SUBJECT_IDENTITY
##                    level CONTROL       PATIENT-DEL   PATIENT-DUP   p      test
##   n                              130           191           64
##   AGE (mean (SD))             14.74 (6.68)  17.17 (7.78)  17.88 (12.40)  0.014
##   SEX (%)          F             64 (49.2)    111 (58.1)     26 (40.6)   0.037
##                    M             66 (50.8)     80 (41.9)     38 (59.4)
```

```
#print(demo_summary, showAllLevels =F)
```

under 35, demographics summary

```
demo_summary_young <- CreateTableOne(data =filter(ucla_demo_tp_agelim, AGE < 35),
 ↪  vars = c("AGE", "SEX"), strata = "SUBJECT_IDENTITY", addOverall =F)
print(demo_summary_young, showAllLevels =T)
```

```
##                   Stratified by SUBJECT_IDENTITY
##                    level CONTROL       PATIENT-DEL   PATIENT-DUP   p      test
##   n                              128           182           54
##   AGE (mean (SD))             14.30 (5.69)  16.06 (6.08)  13.25 (6.33)   0.003
##   SEX (%)          F             62 (48.4)    106 (58.2)     19 (35.2)   0.008
##                    M             66 (51.6)     76 (41.8)     35 (64.8)
```

```
#print(demo_summary, showAllLevels =F)
```

Baseline summary

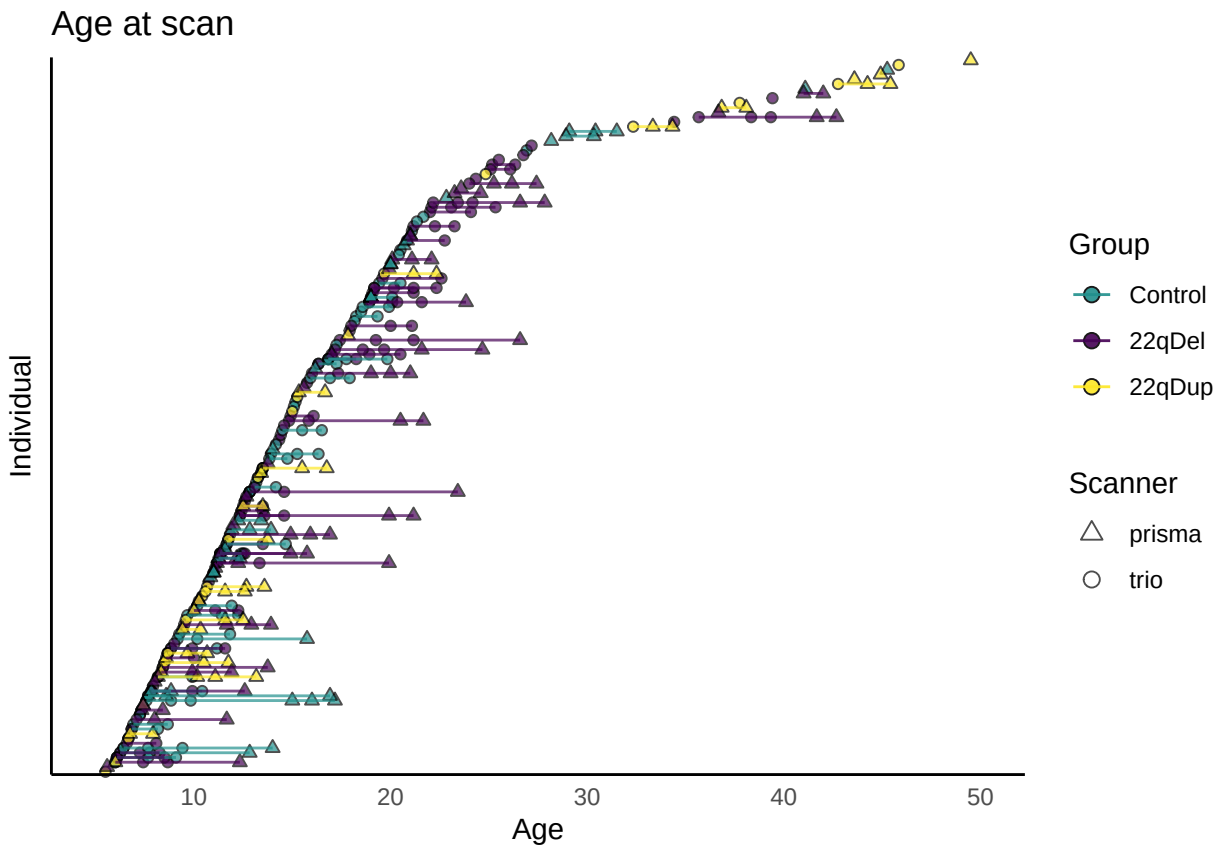
```
demo_summary_bl <- CreateTableOne(data =filter(ucla_demo_tp_agelim, visit_index ==
 ↪  1), vars = c("AGE", "SEX"), strata = "SUBJECT_IDENTITY", addOverall =F)
```

```
print(demo_summary_bl)
```

```
##                   Stratified by SUBJECT_IDENTITY
##                    CONTROL       PATIENT-DEL   PATIENT-DUP   p      test
##   n                   80            96            37
##   AGE (mean (SD)) 14.89 (7.34)  15.52 (7.62)  17.83 (13.50)  0.240
##   SEX = M (%)         39 (48.8)     45 (46.9)     20 (54.1)   0.759
```

Age waterfall plot

```
ucla_demo_waterfall <- ucla_demo_tp_agelim
ucla_demo_waterfall$Scanner <- gsub("-[0-9]", "",
 ↪  ucla_demo_waterfall$converted_timepoint)
ucla_demo_waterfall$Scanner %<>% gsub("T", "trio",.)
ucla_demo_waterfall$Scanner %<>% gsub("P", "prisma",.)
ucla_demo_waterfall$Group <- ucla_demo_waterfall$SUBJECT_IDENTITY
# get age at timepoint 1 for every row
ucla_demo_waterfall$age1 <- lapply(ucla_demo_waterfall$SUBJECTID, function(i)
 ↪  min(filter(ucla_demo_waterfall, SUBJECTID==i)$AGEMONTH)) %>% do.call(rbind,.)

#waterfall <- ggplot(ucla_demo_waterfall, aes(x =(AGEMONTH/12), y =
 ↪  reorder(SUBJECTID, AGEMONTH), color =Group, fill =Group)) +
waterfall <- ggplot(ucla_demo_waterfall, aes(x =(AGEMONTH/12), y = as.factor(age1)
 ↪  , color =Group, fill =Group)) +
  geom_point(aes(shape =Scanner), color = "black", alpha = 0.7) +
  scale_shape_manual(values = c(24,21)) +
  theme_classic() +
  #theme(text=element_text(family = "Arial"))+
  theme(axis.text.y =element_blank(), axis.ticks.x =element_blank(), axis.ticks.y
   ↪  =element_blank()) +
  geom_line(aes(group=SUBJECTID, color =SUBJECT_IDENTITY), alpha = 0.7) +
  scale_fill_manual(values = c("CONTROL" = viridis(3)[2], "PATIENT-DEL" =
   ↪  viridis(3)[1], "PATIENT-DUP" = viridis(3)[3]), labels = c("Control",
   ↪  "22qDel", "22qDup")) +
  scale_color_manual(values = c("CONTROL" = viridis(3)[2], "PATIENT-DEL" =
   ↪  viridis(3)[1], "PATIENT-DUP" = viridis(3)[3]), labels = c("Control",
   ↪  "22qDel", "22qDup")) +
  #guides(color  = guide_legend(order = 1, override.aes = list(color =
   ↪  c(viridis(3)[2], viridis(3)[1], viridis(3)[3])))) +
  guides(fill  = guide_legend(order = 1, override.aes = list(shape = 21, size =
   ↪  2.4, alpha = 0.9, fill = c(viridis(3)[2], viridis(3)[1], viridis(3)[3]))),
         color = guide_legend(order = 1),
         shape = guide_legend(order = 2, override.aes = list(size = 2.5)))+
  ylab("Individual") +
  xlab("Age") +
  xlim(5,50)+
  ggtitle("Age at scan")+
  coord_cartesian(clip = "off")

waterfall
```




```
#ggsave(plot=waterfall, filename =file.path(project,
 ↪  "figures/demographics/waterfall_subcort_age.pdf"), width= 6, height= 6, device
 ↪  = "pdf")
#ggsave(plot=waterfall, filename =file.path(project,
 ↪  "figures/demographics/waterfall_subcort_age.png"), width= 6, height= 6, device
 ↪  = "png")
```

Harmonize sites

merge imaging with demographics

```
# add site column
ucla_demo_tp_agelim$site <- gsub("*-[0-9]", "",
 ↪  ucla_demo_tp_agelim$converted_timepoint)

# merge subcort_all with demo
#demo_sc <- merge(x =ucla_demo_tp_agelim, y =subcort_all, by = "MRI_S_ID", all.x
 ↪  =TRUE)
demo_sc <- merge(x =ucla_demo_tp_agelim, y =subcort_bilat, by = "MRI_S_ID", all.x
 ↪  =TRUE)

# merge with eTIV
demo_sc <- merge(x =demo_sc, y =etiv, by = "MRI_S_ID")
demo_sc$eTIVscaled <- demo_sc$eTIV/1000000

# make numeric gene dosage column from SUBJECT_IDENTITY
demo_sc$gene_dosage <- demo_sc$SUBJECT_IDENTITY %>% gsub("PATIENT-DEL", "1",.) %>%
 ↪  gsub("CONTROL", "2",.) %>% gsub("PATIENT-DUP", "3",.) %>% as.numeric
```

```
# add Age^2
# add age^2
demo_sc$AGE2 <- demo_sc$AGE^2

# update feature names to bilateral columns
sc_names <- names(subcort_bilat)[which(names(subcort_bilat) != "MRI_S_ID")]
```

impute outliers and excluded data (ComBat can't have NAs in input)

```
# list of thalamus regions
#thal_names_final <- names(subcort_all)[grep("Thal", names(subcort_all))]
thal_names_final <- names(subcort_bilat)[grep("Thal", names(subcort_bilat))]
# data frame with MRI_S_ID and region names to remove
# first get all the thalamus ROIs for the subjects with whole thal excluded
remove_thal <- lapply(thal_exclude, function(n) data.frame(MRI_S_ID = n, variable
 ↪  =thal_names_final)) %>% do.call(rbind,.)
# and add that to the subjects/regions in out_filter
#sparse_remove <- rbind(out_filter[, c("MRI_S_ID","variable")], remove_thal)

# add to all subjects/regions with NAs in subcort_bilat
# initialize empty vectors to store row and column names
row_names_true <- character(0)
col_names_true <- character(0)
# prepare input
sc_rn <- is.na(as.data.frame(subcort_bilat)[, sc_names])
rownames(sc_rn) <- subcort_bilat$MRI_S_ID
# loop through the data frame to find TRUE values
for (i in 1:nrow(sc_rn)) {
  for (j in 1:ncol(sc_rn)) {
    if (sc_rn[i, j]) {
      row_names_true <- c(row_names_true, rownames(sc_rn)[i])
      col_names_true <- c(col_names_true, colnames(sc_rn)[j])
    }
  }
}

# Create a data frame from the vectors
bilat_remove <- data.frame(MRI_S_ID = row_names_true, variable = col_names_true)
sparse_remove <- rbind(bilat_remove, remove_thal)

# add site and gene dosage from demo_sc
sparse_remove <- merge(x =sparse_remove, y =demo_sc[, c("MRI_S_ID", "site",
 ↪  "gene_dosage")], by = "MRI_S_ID")

# replace with overall mean for that structure
demo_sc_impute <- demo_sc
for (i in 1:nrow(sparse_remove)){
  sesh <- as.character(sparse_remove[i,"MRI_S_ID"])
  roi <- as.character(sparse_remove[i,"variable"])
  s <- as.character(sparse_remove[i,"site"])
  gd <- as.numeric(sparse_remove[i,"gene_dosage"])
  #print(paste(sesh, roi, s, gd), sep= ", ")
  # first, set to NA in demo_sc_impute
```

```
  demo_sc_impute[which(demo_sc_impute$MRI_S_ID==sesh), roi] <- NA
  # then for that ROI, get mean for the same diagnostic group and scanner as
   ↪  current subject
  demo_filter <- filter(demo_sc_impute, gene_dosage == gd & site == s)
  roi_mean <- mean(demo_filter[, roi], na.rm=TRUE)
  #print(roi_mean)
  # replace with mean
  demo_sc_impute[which(demo_sc_impute$MRI_S_ID==sesh), roi] <- roi_mean
}
```

run longComBat

```
# set up longCombat variables
# formula should match fixed effects in your subsequent analysis
# subject id coded as random effect by (1|subject id variable)
#demovars <- c("MRI_S_ID", "SUBJECTID", "site", "gene_dosage", "AGE", "SEX",
 ↪  "eTIVscaled", "visit_index")
demovars <- c("MRI_S_ID", "SUBJECTID", "site", "gene_dosage", "AGE", "AGE2",
 ↪  "SEX", "eTIVscaled", "visit_index")
features <- sc_names
idvar <- 'MRI_S_ID'
batchvar <- 'site'
timevar <- 'visit_index'
formula <- 'gene_dosage + AGE + AGE2 + SEX + eTIVscaled'
ranef <- '(1|SUBJECTID)'

# make data frame with columns for each variable in the model
# one column for each variable in your formula as well as one column for each
 ↪  neuroimaging feature
# input df should not have any unused columns or package will error
# one row per unique scan
combat_input<- demo_sc_impute[, c(demovars, features)]

# run longCombat
sc_vol_combat <- longCombat(data = combat_input, idvar =idvar, timevar =timevar,
 ↪  batchvar =batchvar, features =features, formula =formula, ranef= ranef,
 ↪  verbose =FALSE)

# get the harmonized data
sc_vol_combat_data <- sc_vol_combat$data_combat

# merge combat back with original
demo_combat <- merge(x =demo_sc, y =sc_vol_combat_data, by = c("MRI_S_ID",
 ↪  "visit_index", "site"))
```

gene dosage and age models

first, set up data frame

```
# get names of combat features
sc_names_combat <- paste0(sc_names,".combat")

# set outliers to NA after combat
demo_combat_na <- demo_combat
```

```
for (i in 1:nrow(sparse_remove)){
  sesh <- as.character(sparse_remove[i,"MRI_S_ID"])
  roi <- as.character(sparse_remove[i,"variable"])
  demo_combat_na[which(demo_combat_na$MRI_S_ID==sesh), paste0(roi,".combat")] <-
 ↪  NA
}

# save demo_combat
#write.csv(demo_combat_na, file =file.path(project,
 ↪  "22q_subcort_vols_combat_na.csv"), quote =TRUE, row.names = FALSE)
```

normalize based on control group mean and SD to get standardized betas

```
# names for normed columns
sc_names_normed <- paste0(sc_names_combat, ".norm")

# for every region, get mean and SD for control group
norm_name_match <- data.frame(combat=sc_names_combat, normed =sc_names_normed,
 ↪  control_mean=NA, control_sd =NA)
for (r in 1:nrow(norm_name_match)){
  region <- norm_name_match[r,"combat"]
  control_dat <- filter(demo_combat_na, SUBJECT_IDENTITY== "CONTROL")[, region]
  norm_name_match[r,"control_mean"] <- mean(control_dat, na.rm=TRUE)
  norm_name_match[r,"control_sd"] <- sd(control_dat, na.rm=TRUE)
}

# df to hold normed data
demo_combat_normed <- demo_combat_na
demo_combat_normed[r, sc_names_normed] <- NA

# for every region in each subject, normalize based on control mean and SD
for (r in 1:nrow(demo_combat_normed)){
  for (c in 1:nrow(norm_name_match)){
    # get column names and control stats for a given region
    combat_name <- norm_name_match[c,"combat"]
    normed_name <- norm_name_match[c,"normed"]
    control_mean <- norm_name_match[c,"control_mean"]
    control_sd <- norm_name_match[c,"control_sd"]
    # get the pre-normed combat-adjusted value
    combat_val <- demo_combat_normed[r, combat_name]
    # if not NA, normalize based on control stats
    if(is.na(combat_val)){
      normed_val <- NA
    }else{
      normed_val <- (combat_val - control_mean)/control_sd
    }
    # store normed value
    demo_combat_normed[r, normed_name] <- normed_val
  }
}

# normalize ICV
td_mean_icv <- filter(demo_combat_normed, gene_dosage == 2)$eTIV %>% mean
td_sd_icv <- filter(demo_combat_normed, gene_dosage == 2)$eTIV %>% sd
```

```
demo_combat_normed$eTIVnormed <- (demo_combat_normed$eTIV - td_mean_icv)/td_sd_icv
```

longitudinal demo table

```
#dir <- "/Users/charlie/Dropbox/PhD/bearden_lab/22q/analyses/striatum_thalamus_fc"

# get variables from demo_mri
df_demo <- demo_combat_na

### hand
# get handedness item scores coded in sistat as 1=L, 2=R, 3=either, 0=no
 ↪  experience
edin <- df_all_ucla$edin[, c("SUBJECTID", "CONVERTEDVISITNUM", "EDIN1", "EDIN2",
 ↪  "EDIN3", "EDIN4", "EDIN5", "EDIN6", "EDIN7", "EDIN8", "EDIN9", "EDIN10")]
# function to get total edinburgh score and handedness
# formula is 100*(R-L)/(R+L). score < -40 means left handed, score > 40 right
 ↪  handed
# if more than 2 items NA then score is NA
get_hand <- function(edin){
  sub <- edin[c("SUBJECTID")]
  visit <- edin[c("CONVERTEDVISITNUM")]
  scores <- edin[c("EDIN1", "EDIN2", "EDIN3", "EDIN4", "EDIN5", "EDIN6", "EDIN7",
 ↪  "EDIN8", "EDIN9", "EDIN10")]
  l <- sum(scores == 1, na.rm=TRUE)
  r <- sum(scores == 2, na.rm=TRUE)
  na <- sum(is.na(scores))
  if (na < 3){
    score <- 10*(r-l)
  }
  if (na > 2){
    hand <- NA
    score <- NA
  }else if(score > 40){
    hand <-"R"
  }else if (score < -40){
    hand <- "L"
  }else if (score >= -40 & score <= 40) {
    hand <- "A"
  }else{
    hand <- NA
  }
  output <- cbind(sub, visit, score, hand) %>% as.data.frame
  colnames(output) <- c("SUBJECTID", "CONVERTEDVISITNUM", "hand_score", "hand")
  return(output)
}
# get handedness
edin_result <- lapply(1:nrow(edin), function(r) get_hand(edin[r,])) %>%
 ↪  do.call(rbind,.) %>% as.data.frame

# merge handedness with demo table
df_demo <- merge(x =df_demo, y =edin_result[c("SUBJECTID", "CONVERTEDVISITNUM",
 ↪  "hand")], by = c("SUBJECTID", "CONVERTEDVISITNUM"), all.x =T)

# manually fix a few subjects' handedness
```

```r
#q_0017= "A"
df_demo[which(df_demo$SUBJECTID == "q_0017"),"hand"] <- "A"
#q_0263= "R"
df_demo[which(df_demo$SUBJECTID == "q_0263"),"hand"] <- "R"
#q_0331= "R"
df_demo[which(df_demo$SUBJECTID == "q_0331"),"hand"] <- "R"

### psych dx
# first get SCID columns with Dx (currently only using patient Dx not collateral)
scid_dx_all <- df_all_ucla$SCID[, c("PATCODE1", "PATCODE2", "PATCODE3",
 ↪  "PATCODE4", "PATCODE5", "PATCODE6", "PATCODE7", "PATCODE8")]

# get list of unique dx entries
dx_unique <- scid_dx_all %>% as.matrix %>% as.vector %>% sort %>% unique

# create matching key between unique dx and dx groups for demographics table
# first save dx_unique as csv
#write.table(dx_unique, file =file.path(csvdir_ucla,"scid_unique_dx.csv"),
 ↪  row.names =F, col.names =F)
# then manually edit csv so that column 2 contains the dx group for each specific
 ↪  dx. save edited csv as scid_unique_dx_matching.csv
# dx group categories based on DSM-5
 ↪  https://www.psychiatry.org/File%20Library/Psychiatrists/Practice/DSM/APA_DSM-5-Contents.pdf
# Notes: leave second column blank for non-psych dx (eg Crohn's), code
 ↪  single-episode MDD in full remission as depressive_disorder_past, all other
 ↪  MDD as depressive_disorder
# read matching table back in
dx_unique_matching <- read.csv(file =file.path(project,
 ↪  "demographics/scid_unique_dx_matching.csv"), header =F)

# function to take scid patient codes 1-8 for a subject and output binary y/n for
 ↪  each dx in dx_groups based on dx_unique_matching
# should be applied to rows of the scid data frame
get_general_dx_scid <- function(scid_row, dx_matching){
  # get subject id and visit columns
  id_cols <- scid_row[c("SUBJECTID", "CONVERTEDVISITNUM")]
  # get list of all unique dx groups in matching key
  dx_groups <- dx_matching[,2] %>% sort %>% unique
  dx_groups <- dx_groups[dx_groups != ""]
  # get patcodes 1-8
  patcodes_all <- scid_row[c("PATCODE1", "PATCODE2", "PATCODE3", "PATCODE4",
 ↪  "PATCODE5", "PATCODE6", "PATCODE7", "PATCODE8")] %>% as.matrix
  patcodes <- patcodes_all[patcodes_all != ""]
  # if subject has data in patcodes, convert to dx groups
  if(length(patcodes) > 0){
    # get dx group for each patcode by referencing dx_matching
    sub_dx <- lapply(patcodes, function(x) filter(dx_matching, V1 == x)$V2) %>%
 ↪  do.call(cbind,.) %>% as.matrix
    sub_dx <- sub_dx[sub_dx != ""]
    # check if subject has SCID dx in each dx group, return TRUE for yes, FALSE
     ↪  for no
    dx_yn <- lapply(dx_groups, function(x) x %in% sub_dx) %>% do.call(cbind,.) %>%
 ↪  as.data.frame
```

```
  # return empty columns if no patcode data (without this will fail for subjects
   ↪  with no data)
  }else{
    dx_yn <- matrix(nrow= 1, ncol =length(dx_groups)) %>% as.data.frame
  }
  colnames(dx_yn) <- paste("SCID", dx_groups, sep= "_")
  output <- cbind(id_cols, dx_yn)
  return(output)
}

# get general dx for each scid entry
scid_general <- lapply(1:nrow(df_all_ucla$SCID), function(r)
 ↪  get_general_dx_scid(scid_row=df_all_ucla$SCID[r,],
 ↪  dx_matching=dx_unique_matching)) %>% do.call(rbind,.) %>% as.data.frame

# merge scid general with demo table
df_demo <- merge(x =df_demo, y =scid_general, by = c("SUBJECTID",
 ↪  "CONVERTEDVISITNUM"), all.x =T)

# count instances of each dx
dx_counts <- df_demo %>% dplyr::select(starts_with("SCID_")) %>% colSums(na.rm=T)

# get list of dx with more than 2 instances in the data set
dx_use <- which(dx_counts > 2) %>% names

# remove depressive_disorder_past (single episode full remission)
dx_use <- dx_use[dx_use != "SCID_Depression_Related_Past"]
# remove learning disorder
dx_use <- dx_use[dx_use != "SCID_Learning_Disorder"]

# add info from summPsych
summpsych <- df_all_ucla$summPsych

# meds as factors
summpsych$PSYTYPE <- factor(summpsych$PSYTYPE, levels = c(1,2,3,4,5), labels =
 ↪  c("antipsychotic", "antidepressant_or_mood_stabilizer", "stimulant", "other",
 ↪  "none"))

# merge meds with demo table
df_demo <- merge(x =df_demo, y =summpsych[, c("SUBJECTID", "CONVERTEDVISITNUM",
 ↪  "PSYTYPE")], by = c("SUBJECTID", "CONVERTEDVISITNUM"), all.x =T) %>%
 ↪  rename("psych_meds" = "PSYTYPE")

# function to mark subject as ASD positive if positive at any visit in summPsych
get_asd <- function(subject, summ_psych){
  sp_all <- filter(summ_psych, summ_psych$SUBJECTID==subject)
  if(nrow(sp_all)>0){
    asd_all <- sp_all$ASDDIAGNOS
    # check if any visit coded as 1 (meaning asd = yes)
    asd_yn <- (1 %in% asd_all)
  }else{
    asd_yn <- NA
  }
```

```
  return(asd_yn)
}

# add ASD column based on summPsych
asd_col <- lapply(1:nrow(df_demo), function(r)
 ↪  get_asd(subject=df_demo[r,"SUBJECTID"], summ_psych=summpsych)) %>%
 ↪  do.call(rbind,.) %>% as.data.frame
colnames(asd_col) <- "summPsych_ASD"

# merge summPsych ASD with demo table
df_demo <- cbind(df_demo, asd_col)

# remove SCID_ASD column, redundant with summPsych
dx_use <- dx_use[dx_use != "SCID_ASD"]

# function to get psychosis status from SIPS
# to be applied to the row indices of a demographics df, and also given the SIPS
 ↪  df
get_sips <- function(r, demo, sips){
   sub <- demo$SUBJECTID[[r]]
   visit <- demo$CONVERTEDVISITNUM[[r]]
   df_out <- data.frame(SUBJECTID=sub, CONVERTEDVISITNUM=visit)
   sips_sesh <- sips %>% filter(SUBJECTID == sub & CONVERTEDVISITNUM == visit)
   if(nrow(sips_sesh) < 1){
     # if no match for sub+visit in sips table, set outputs to NA
     df_out[, c("SIPS_p_sum", "SIPS_n_sum", "SIPS_d_sum", "SIPS_g_sum",
 ↪  "SIPS_total", "SIPS_psychosis_6", "SIPS_psspectrum_3")] <- rep(NA, times = 7)
   }else if(nrow(sips_sesh) > 1){
     # if more than one match for sub+visit, note error
     df_out[, c("SIPS_p_sum", "SIPS_n_sum", "SIPS_d_sum", "SIPS_g_sum",
 ↪  "SIPS_total", "SIPS_psychosis_6", "SIPS_psspectrum_3")] <-
 ↪  rep("ERROR-duplicates", times = 7)
   }else{
     # get SIPS P scores
     sips_p_scores <- sips_sesh[c("P1SEV", "P2SEV", "P3SEV", "P4SEV", "P5SEV")]
     # sum SIPS P
     df_out[,"SIPS_p_sum"] <- sum(sips_p_scores)
     # get SIPS N
     sips_n_scores <- sips_sesh[c("N1SEV", "N2SEV", "N3SEV", "N4SEV", "N5SEV",
 ↪  "N6SEV")]
     df_out[,"SIPS_n_sum"] <- sum(sips_n_scores)
     # get SIPS D
     sips_d_scores <- sips_sesh[c("D1SEV", "D2SEV", "D3SEV", "D4SEV")]
     df_out[,"SIPS_d_sum"] <- sum(sips_d_scores)
     # get SIPS G
     sips_g_scores <- sips_sesh[c("G1SEV", "G2SEV", "G3SEV", "G4SEV")]
     df_out[,"SIPS_g_sum"] <- sum(sips_g_scores)
     # get SIPS total
     df_out["SIPS_total"] <- (df_out["SIPS_p_sum"] +  df_out["SIPS_n_sum"] +
 ↪  df_out["SIPS_d_sum"] +  df_out["SIPS_g_sum"] )
     # check psychosis criteria of at least one SIPS P score of 6
     count_6 <- length(which(sips_p_scores == 6))
     if(is.na(sum(sips_p_scores))){
```

```
      df_out[,"SIPS_psychosis_6"] <- NA
    }else if(count_6 > 0){
      df_out[,"SIPS_psychosis_6"] <- 1
    }else{
      df_out[,"SIPS_psychosis_6"] <- 0
    }
    # check psychosis-spectrum criteria of at least one SIPS P >= 3
    count_3 <- length(which(sips_p_scores >= 3))
    if(is.na(sum(sips_p_scores))){
      df_out[,"SIPS_psspectrum_3"] <- NA
    }else if(count_3 > 0){
      df_out[,"SIPS_psspectrum_3"] <- 1
    }else{
      df_out[,"SIPS_psspectrum_3"] <- 0
    }
  }
  return(df_out)
}

# get sips
demo_table_sips <- lapply(1:nrow(df_demo), function(r) get_sips(r = r,
 ↪  demo=df_demo, sips =df_all_ucla$SIPS)) %>% do.call(rbind,.)

# merge sips with demo table
df_demo <- merge(x =df_demo, y =demo_table_sips[, c("SUBJECTID",
 ↪  "CONVERTEDVISITNUM", "SIPS_total", "SIPS_psspectrum_3", "SIPS_p_sum")], by =
 ↪  c("SUBJECTID", "CONVERTEDVISITNUM"), all.x =T) %>% rename("SIPS_prodromal" =
 ↪  "SIPS_psspectrum_3")

# set sips_total to numeric and sips_prodromal to factor
df_demo %<>% mutate_at(vars("SIPS_prodromal"), ~as.logical(.))
df_demo %<>% mutate_at(vars("SIPS_total", "SIPS_p_sum"), ~as.numeric(.))

# get IQ
# WASI, WISC-IV, DKEFS and trail making all under df_all$DKEFS for trio data
# IQSS -- full scale WASI
ucla_neuro1 <- df_all_ucla$DKEFS[, c("SUBJECTID", "CONVERTEDVISITNUM", "VOCASS",
 ↪  "MATRIXSS", "IQSS")] %>% rename("WASI_verbal" = "VOCASS") %>%
 ↪  rename("WASI_matrix" = "MATRIXSS") %>% rename("IQ_full" = "IQSS")
# renewal neuro (prisma) under df_all$neurocogTest
ucla_neuro2 <- df_all_ucla$neurocogTest[, c("SUBJECTID", "CONVERTEDVISITNUM",
 ↪  "VOCA_TSCORE", "MATRIX_TSCORE", "IQ_SCORE")] %>% rename("WASI_verbal" =
 ↪  "VOCA_TSCORE") %>% rename("WASI_matrix" = "MATRIX_TSCORE") %>%
 ↪  rename("IQ_full" = "IQ_SCORE")
# combine 22q orig and renewal scores before merging with demo table
ucla_neuro <- rbind(ucla_neuro1, ucla_neuro2)
# merge neuro with demo table
df_demo <- merge(x =df_demo, y =ucla_neuro[, c("SUBJECTID", "CONVERTEDVISITNUM",
 ↪  "IQ_full", "WASI_verbal", "WASI_matrix")], by = c("SUBJECTID",
 ↪  "CONVERTEDVISITNUM"), all.x =T)
# record IQ instrument
df_demo$IQ_measure <- NA
df_demo$IQ_measure[!is.na(df_demo$IQ_full)] <- "WASI_full_scale"
```

```
# get subjects missing TESTDATE
missing_date <- filter(demo_combat_na, is.na(TESTDATE))

# manually add missing dates
demo_combat_na[which(demo_combat_na$MRI_S_ID== "Q_0192_06172014"),"TESTDATE"] <-
 ↪  "06/17/2014"
demo_combat_na[which(demo_combat_na$MRI_S_ID== "Q_0508_06232022"),"TESTDATE"] <-
 ↪  "06/23/2022"
demo_combat_na[which(demo_combat_na$MRI_S_ID== "Q_0519_05312022"),"TESTDATE"] <-
 ↪  "05/31/2022"
demo_combat_na[which(demo_combat_na$MRI_S_ID== "Q_0520_06012022"),"TESTDATE"] <-
 ↪  "06/01/2022"
demo_combat_na[which(demo_combat_na$MRI_S_ID== "Q_0521_05202022"),"TESTDATE"] <-
 ↪  "05/20/2022"
demo_combat_na[which(demo_combat_na$MRI_S_ID== "Q_0525_06072022"),"TESTDATE"] <-
 ↪  "06/07/2022"
demo_combat_na[which(demo_combat_na$MRI_S_ID== "Q_0526_06242022"),"TESTDATE"] <-
 ↪  "06/24/2022"
demo_combat_na[which(demo_combat_na$MRI_S_ID== "Q_0527_07112022"),"TESTDATE"] <-
 ↪  "07/11/2022"
demo_combat_na[which(demo_combat_na$MRI_S_ID== "Q_0528_07202022"),"TESTDATE"] <-
 ↪  "07/20/2022"
demo_combat_na[which(demo_combat_na$MRI_S_ID== "Q_0529_07202022"),"TESTDATE"] <-
 ↪  "07/20/2022"
demo_combat_na[which(demo_combat_na$MRI_S_ID== "Q_0561_11032022"),"TESTDATE"] <-
 ↪  "11/03/2022"
demo_combat_na[which(demo_combat_na$MRI_S_ID== "Q_0568_10252022"),"TESTDATE"] <-
 ↪  "10/25/2022"

# vector of variables for demo table
#vars_use <- c("AGE", "SEX", "EDUDAD", "EDUMOM", "EDUYEARS", "hand",
 ↪  "percent_BOLD_scrubbed", "IQ_full", "WASI_verbal", "WASI_matrix",
 ↪  "SIPS_total", "SIPS_prodromal", dx_use, "summPsych_ASD", "psych_meds")
vars_use <- c("AGE", "SEX", "hand", "IQ_full", "SIPS_total", "SIPS_p_sum",
 ↪  "SIPS_prodromal", dx_use, "summPsych_ASD", "psych_meds")

# make table
demo_match_final <- CreateTableOne(data =df_demo, vars =vars_use, strata =
 ↪  "SUBJECT_IDENTITY", addOverall =F, includeNA=T)
demo_match_final

# export tableone
export_demo_table <- print(demo_match_final, quote =F, noSpaces =F, printToggle
 ↪  =T)
#write.csv(export_demo_table, file =file.path(project, "table1_demographics.csv"))
```

final baseline demo table note: some interim variable names re-used from longitudinal table but final df_demo_table and df_demo_table_bl should be accurate)

```
#dir <- "/Users/charlie/Dropbox/PhD/bearden_lab/22q/analyses/striatum_thalamus_fc"

# get variables from demo_mri
#df_demo_table_bl <- filter(demo_combat, visit_index == 1)[, c("SUBJECTID",
 ↪  "CONVERTEDVISITNUM", "MRI_S_ID", "SUBJECT_IDENTITY", "AGE", "SEX", "EDUDAD",
 ↪  "EDUMOM", "EDUYEARS")]
```

```
# all columns, no NA
# TODO: impute for plsr?
#df_demo_table_bl <- filter(demo_combat_na, visit_index == 1)
df_demo_table_bl <- filter(demo_combat, visit_index == 1)

### hand
# get handedness item scores coded in sistat as 1=L, 2=R, 3=either, 0=no
 ↪  experience
edin <- df_all_ucla$edin[, c("SUBJECTID", "CONVERTEDVISITNUM", "EDIN1", "EDIN2",
 ↪  "EDIN3", "EDIN4", "EDIN5", "EDIN6", "EDIN7", "EDIN8", "EDIN9", "EDIN10")]
edin_result <- lapply(1:nrow(edin), function(r) get_hand(edin[r,])) %>%
 ↪  do.call(rbind,.) %>% as.data.frame

# merge handedness with demo table
df_demo_table_bl <- merge(x =df_demo_table_bl, y =edin_result[c("SUBJECTID",
 ↪  "CONVERTEDVISITNUM", "hand")], by = c("SUBJECTID", "CONVERTEDVISITNUM"), all.x
 ↪  =T)

# manually fix a few subjects' handedness
#q_0017= "A"
df_demo_table_bl[which(df_demo_table_bl$SUBJECTID == "q_0017"),"hand"] <- "A"
#q_0263= "R"
df_demo_table_bl[which(df_demo_table_bl$SUBJECTID == "q_0263"),"hand"] <- "R"
#q_0331= "R"
df_demo_table_bl[which(df_demo_table_bl$SUBJECTID == "q_0331"),"hand"] <- "R"

### psych dx
# first get SCID columns with Dx (currently only using patient Dx not collateral)
scid_dx_all <- df_all_ucla$SCID[, c("PATCODE1", "PATCODE2", "PATCODE3",
 ↪  "PATCODE4", "PATCODE5", "PATCODE6", "PATCODE7", "PATCODE8")]

# get list of unique dx entries
dx_unique <- scid_dx_all %>% as.matrix %>% as.vector %>% sort %>% unique

# create matching key between unique dx and dx groups for demographics table
# first save dx_unique as csv
#write.table(dx_unique, file =file.path(csvdir_ucla,"scid_unique_dx.csv"),
 ↪  row.names =F, col.names =F)
# then manually edit csv so that column 2 contains the dx group for each specific
 ↪  dx. save edited csv as scid_unique_dx_matching.csv
# dx group categories based on DSM-5
 ↪  https://www.psychiatry.org/File%20Library/Psychiatrists/Practice/DSM/APA_DSM-5-Contents.pdf
# Notes: leave second column blank for non-psych dx (eg Crohn's), code
 ↪  single-episode MDD in full remission as depressive_disorder_past, all other
 ↪  MDD as depressive_disorder
# read matching table back in
dx_unique_matching <- read.csv(file =file.path(project,
 ↪  "demographics/scid_unique_dx_matching.csv"), header =F)

# get general dx for each scid entry
scid_general <- lapply(1:nrow(df_all_ucla$SCID), function(r)
 ↪  get_general_dx_scid(scid_row=df_all_ucla$SCID[r,],
 ↪  dx_matching=dx_unique_matching)) %>% do.call(rbind,.) %>% as.data.frame
```

```r
# merge scid general with demo table
df_demo_table_bl <- merge(x =df_demo_table_bl, y =scid_general, by =
 ↪  c("SUBJECTID", "CONVERTEDVISITNUM"), all.x =T)

# count instances of each dx
dx_counts <- df_demo_table_bl %>% dplyr::select(starts_with("SCID_")) %>%
 ↪  colSums(na.rm=T)

# get list of dx with more than 2 instances in the data set
dx_use <- which(dx_counts > 2) %>% names

# remove depressive_disorder_past (single episode full remission)
dx_use <- dx_use[dx_use != "SCID_Depression_Related_Past"]
# remove learning disorder
dx_use <- dx_use[dx_use != "SCID_Learning_Disorder"]

# add info from summPsych
summpsych <- df_all_ucla$summPsych

# meds as factors
summpsych$PSYTYPE <- factor(summpsych$PSYTYPE, levels = c(1,2,3,4,5), labels =
 ↪  c("antipsychotic", "antidepressant_or_mood_stabilizer", "stimulant", "other",
 ↪  "none"))

# merge meds with demo table
df_demo_table_bl <- merge(x =df_demo_table_bl, y =summpsych[, c("SUBJECTID",
 ↪  "CONVERTEDVISITNUM", "PSYTYPE")], by = c("SUBJECTID", "CONVERTEDVISITNUM"),
 ↪  all.x =T) %>% rename("psych_meds" = "PSYTYPE")

# add ASD column based on summPsych
asd_col <- lapply(1:nrow(df_demo_table_bl), function(r)
 ↪  get_asd(subject=df_demo_table_bl[r,"SUBJECTID"], summ_psych=summpsych)) %>%
 ↪  do.call(rbind,.) %>% as.data.frame
colnames(asd_col) <- "summPsych_ASD"

# merge summPsych ASD with demo table
df_demo_table_bl <- cbind(df_demo_table_bl, asd_col)

# remove SCID_ASD column, redundant with summPsych
dx_use <- dx_use[dx_use != "SCID_ASD"]

# add IQ and merge with demo table
# TO-DO figure out neuropsych test date matching
#neurocog <- df_all$neurocogTest[, c("SUBJECTID", "CONVERTEDVISITNUM", "IQ_SCORE",
 ↪  "VOCA_TSCORE", "MATRIX_TSCORE")]
#df_demo_table_full <- merge(x =df_demo_table_full, y =neurocog, by =
 ↪  c("SUBJECTID", "CONVERTEDVISITNUM"), all.x =T)

# get sips
demo_table_sips <- lapply(1:nrow(df_demo_table_bl), function(r) get_sips(r = r,
 ↪  demo=df_demo_table_bl, sips =df_all_ucla$SIPS)) %>% do.call(rbind,.)
```

```
# merge sips with demo table
df_demo_table_bl <- merge(x =df_demo_table_bl, y =demo_table_sips[, c("SUBJECTID",
 ↪  "CONVERTEDVISITNUM", "SIPS_total", "SIPS_psspectrum_3", "SIPS_p_sum")], by =
 ↪  c("SUBJECTID", "CONVERTEDVISITNUM"), all.x =T) %>% rename("SIPS_prodromal" =
 ↪  "SIPS_psspectrum_3")
# set sips_total to numeric and sips_prodromal to factor
df_demo_table_bl %<>% mutate_at(vars("SIPS_prodromal"), ~as.logical(.))
df_demo_table_bl %<>% mutate_at(vars("SIPS_total"), ~as.numeric(.))
df_demo_table_bl %<>% mutate_at(vars("SIPS_p_sum"), ~as.numeric(.))

# also merge full SIPS codes
sips_sev <- grep("SEV", names(df_all_ucla$SIPS), value =TRUE)
df_demo_table_bl <- merge(x =df_demo_table_bl, y =df_all_ucla$SIPS[,
 ↪  cbind(sips_sev, "SUBJECTID", "CONVERTEDVISITNUM")], by = c("SUBJECTID",
 ↪  "CONVERTEDVISITNUM"), all.x =TRUE)

# get SRS categories
srs_ts <- c("TSRAW", "TSAWARE", "TSCOGNIT", "TSCOMMUN", "TSMOTIV", "TSAUTIST")
df_demo_table_bl <- merge(x =df_demo_table_bl, y =df_all_ucla$SRS[, cbind(srs_ts,
 ↪  "SUBJECTID", "CONVERTEDVISITNUM")], by = c("SUBJECTID", "CONVERTEDVISITNUM"),
 ↪  all.x =TRUE)
# get SRS items
srs_items <- grep("SRS", names(df_all_ucla$SRS), value =TRUE)
df_demo_table_bl <- merge(x =df_demo_table_bl, y =df_all_ucla$SRS[,
 ↪  cbind(srs_items, "SUBJECTID", "CONVERTEDVISITNUM")], by = c("SUBJECTID",
 ↪  "CONVERTEDVISITNUM"), all.x =TRUE)

# get IQ
# WASI, WISC-IV, DKEFS and trail making all under df_all$DKEFS for trio data
# IQSS -- full scale WASI
ucla_neuro1 <- df_all_ucla$DKEFS[, c("SUBJECTID", "CONVERTEDVISITNUM", "VOCASS",
 ↪  "MATRIXSS", "IQSS")] %>% rename("WASI_verbal" = "VOCASS") %>%
 ↪  rename("WASI_matrix" = "MATRIXSS") %>% rename("IQ_full" = "IQSS")
# renewal neuro (prisma) under df_all$neurocogTest
ucla_neuro2 <- df_all_ucla$neurocogTest[, c("SUBJECTID", "CONVERTEDVISITNUM",
 ↪  "VOCA_TSCORE", "MATRIX_TSCORE", "IQ_SCORE")] %>% rename("WASI_verbal" =
 ↪  "VOCA_TSCORE") %>% rename("WASI_matrix" = "MATRIX_TSCORE") %>%
 ↪  rename("IQ_full" = "IQ_SCORE")
# combine 22q orig and renewal scores before merging with demo table
ucla_neuro <- rbind(ucla_neuro1, ucla_neuro2)
# merge neuro with demo table
df_demo_table_bl <- merge(x =df_demo_table_bl, y =ucla_neuro[, c("SUBJECTID",
 ↪  "CONVERTEDVISITNUM", "IQ_full", "WASI_verbal", "WASI_matrix")], by =
 ↪  c("SUBJECTID", "CONVERTEDVISITNUM"), all.x =T)
# record IQ instrument
df_demo_table_bl$IQ_measure <- NA
df_demo_table_bl$IQ_measure[!is.na(df_demo_table_bl$IQ_full)] <- "WASI_full_scale"

# visit counts
# get number of longitudinal visits per subject
# apply to list of subjects ids in baseline df
get_n_visits <- function(subject, full_df){
  sub_all <- filter(full_df, SUBJECTID == subject)
```

```
  n_sesh <- nrow(sub_all)
  return(n_sesh)
}
df_demo_table_bl$visit_counts <- lapply(df_demo_table_bl$SUBJECTID, function(s)
 ↪  get_n_visits(subject=s, full_df=demo_combat)) %>% do.call(rbind,.) %>%
 ↪  as.vector
df_demo_table_bl$visit_counts %<>% as.numeric

# get subjects missing TESTDATE
missing_date <- filter(demo_combat_na, is.na(TESTDATE))

# manually add missing dates
demo_combat_na[which(demo_combat_na$MRI_S_ID== "Q_0192_06172014"),"TESTDATE"] <-
 ↪  "06/17/2014"
demo_combat_na[which(demo_combat_na$MRI_S_ID== "Q_0508_06232022"),"TESTDATE"] <-
 ↪  "06/23/2022"
demo_combat_na[which(demo_combat_na$MRI_S_ID== "Q_0519_05312022"),"TESTDATE"] <-
 ↪  "05/31/2022"
demo_combat_na[which(demo_combat_na$MRI_S_ID== "Q_0520_06012022"),"TESTDATE"] <-
 ↪  "06/01/2022"
demo_combat_na[which(demo_combat_na$MRI_S_ID== "Q_0521_05202022"),"TESTDATE"] <-
 ↪  "05/20/2022"
demo_combat_na[which(demo_combat_na$MRI_S_ID== "Q_0525_06072022"),"TESTDATE"] <-
 ↪  "06/07/2022"
demo_combat_na[which(demo_combat_na$MRI_S_ID== "Q_0526_06242022"),"TESTDATE"] <-
 ↪  "06/24/2022"
demo_combat_na[which(demo_combat_na$MRI_S_ID== "Q_0527_07112022"),"TESTDATE"] <-
 ↪  "07/11/2022"
demo_combat_na[which(demo_combat_na$MRI_S_ID== "Q_0528_07202022"),"TESTDATE"] <-
 ↪  "07/20/2022"
demo_combat_na[which(demo_combat_na$MRI_S_ID== "Q_0529_07202022"),"TESTDATE"] <-
 ↪  "07/20/2022"
demo_combat_na[which(demo_combat_na$MRI_S_ID== "Q_0561_11032022"),"TESTDATE"] <-
 ↪  "11/03/2022"
demo_combat_na[which(demo_combat_na$MRI_S_ID== "Q_0568_10252022"),"TESTDATE"] <-
 ↪  "10/25/2022"

# visit interval
# get inter-visit interval
# apply to list of subjects ids in baseline df
get_avg_interval <- function(subject, full_df){
  dates_initial <- filter(full_df, SUBJECTID == subject)$TESTDATE
  dates_all <- sort(as.Date(dates_initial, format= "%m/%d/%Y"))
  intervals<-NULL
  ndates<-length(dates_all)
  # NA if only one visit
  if(ndates == 1){
    intervals<-NA
    avg_interval<-NA
  }else if (ndates>1){
    # starting with the second visit, get all intervals between date i and i-1
    for (i in 2:ndates){
      #diff <- difftime(as.Date(dates_all[i], format= "%m/%d/%Y" ),
       ↪  as.Date(dates_all[i-1], format= "%m/%d/%Y"), units = "days") %>%
       ↪  as.numeric
```

```
      diff <- difftime(dates_all[i], dates_all[i-1], units = "days") %>%
 ↪  as.numeric
      intervals <- c(intervals, diff)
    }
    avg_interval <- mean(intervals)
  }
  return(avg_interval)
}
df_demo_table_bl$avg_interval <- lapply(df_demo_table_bl$SUBJECTID, function(s)
 ↪  get_avg_interval(subject=s, full_df=demo_combat_na)) %>% do.call(rbind,.) %>%
 ↪  as.vector

# vector of variables for demo table
#vars_use <- c("AGE", "SEX", "EDUDAD", "EDUMOM", "EDUYEARS", "hand",
 ↪  "percent_BOLD_scrubbed", "IQ_full", "WASI_verbal", "WASI_matrix",
 ↪  "SIPS_total", "SIPS_prodromal", dx_use, "summPsych_ASD", "psych_meds")
vars_use <- c("AGE", "SEX", "hand", "IQ_full", "SIPS_total", "SIPS_p_sum",
 ↪  "SIPS_prodromal", dx_use, "summPsych_ASD", "psych_meds", "visit_counts",
 ↪  "avg_interval")

# make table
demo_match_final_bl <- CreateTableOne(data =df_demo_table_bl, vars =vars_use,
 ↪  strata = "SUBJECT_IDENTITY", addOverall =F, includeNA=T)
demo_match_final_bl

# export tableone
export_demo_table_bl <- print(demo_match_final_bl, quote =F, noSpaces =F,
 ↪  printToggle =T)

#export_demo_table
#write.csv(export_demo_table_bl, file =file.path(project,
 ↪  "/figures/demographics/table1_demographics.csv"))
```

make demo table for export

```
# read manually edited demo table back in
demo_edit <- read.csv(file.path(project,
 ↪  "figures/demographics/table1_demographics_edit.csv")) %>% rename(" "= "X") %>%
 ↪  rename("22qDel"= "X22qDel") %>% rename("22qDup"= "X22qDup") %>%
 ↪  rename("p-value"= "p.value")
#demo_edit$`p-value` %<>% as.numeric %>% formatC(., format = "g", digits = 2)

# add info on scanner by visit
get_scanner_by_visit <- function(df_long=df_demo, visit){
  df <- filter(df_long, visit_index ==visit)
  t <- CreateTableOne(data =df, vars = "site", strata = "SUBJECT_IDENTITY")
  out <- print(t, showAllLevels =TRUE)["site (%)",]
  return(out)
}
scanner_visit <- lapply(1:6, function(i) get_scanner_by_visit(visit=i)) %>%
 ↪  do.call(rbind,.) %>% as.data.frame
scanner_visit_final <- scanner_visit
scanner_visit_final[1:6,1] <- paste("Visit",1:6, "Prisma scanner, n (%)")
colnames(scanner_visit_final) <- colnames(demo_edit)
```

```
demo_edit_final <- rbind(demo_edit, scanner_visit_final[1:6,1:5])
```

```
# export table
demo_out <- demo_edit_final %>% gt() %>%
  tab_style(style = cell_text(weight = "bold"), locations =
   ↪  list(cells_body(columns = 1, rows = 1:nrow(demo_edit_final)),
   ↪  cells_column_labels())) %>%
  cols_align(align= "right", columns =everything())
demo_out
```

	TD	22qDel	22qDup	p-value
n	80	96	37	
Age, mean (SD)	14.89 (7.34)	15.52 (7.62)	17.83 (13.50)	0.24
Sex, n (%) Female	41 (51.3)	51 (53.1)	17 (45.9)	0.759
Full Scale IQ, mean (SD)	111.27 (19.28)	78.65 (12.74)	95.44 (17.84)	<0.001
SIPS Positive total, mean (SD)	1.23 (1.88)	5.86 (6.52)	2.96 (3.25)	<0.001
Psychosis Risk Symptoms, n (%)	4 (5.0)	24 (25.0)	5 (13.5)	0.002
Psychotic Disorder, n (%)	0 (0.0)	8 (8.3)	0 (0.0)	0.022
ADHD, n (%)	5 (6.2)	41 (42.7)	14 (37.8)	<0.001
Autism, n (%)	0 (0.0)	45 (46.9)	15 (40.5)	<0.001
Antipsychotic Med, n (%)	0 (0.0)	11 (11.5)	2 (5.4)	<0.001
Visit count, mean (SD)	1.62 (0.89)	1.99 (1.16)	1.73 (0.93)	0.058
Days between visits, mean (SD)	667.68 (546.90)	676.78 (383.58)	483.15 (111.84)	0.26
Visit 1 Prisma scanner, n (%)	25 (31.2)	23 (24.0)	16 (43.2)	0.090
Visit 2 Prisma scanner, n (%)	8 (22.9)	16 (29.6)	16 (100.0)	<0.001
Visit 3 Prisma scanner, n (%)	4 (36.4)	13 (46.4)	10 (100.0)	0.005
Visit 4 Prisma scanner, n (%)	2 (100.0)	11 (100.0)	1 (100.0)	NA
Visit 5 Prisma scanner, n (%)	1 (100.0)	4 (100.0)	-	NA
Visit 6 Prisma scanner, n (%)	1 (100.0)	-	-	NA

```
# gtsave(demo_out, filename = file.path(project,
 ↪  "figures/demographics/table1.pdf"))
# gtsave(demo_out, filename = file.path(project,
 ↪  "figures/demographics/table1.png"))
# gtsave(demo_out, filename = file.path(project,
 ↪  "figures/demographics/table1.rtf"))
# gtsave(demo_out, filename = file.path(project,
 ↪  "figures/demographics/table1.tex"))
```

GAMMs

GAMMs in at every ROI with a factor-smooth term for age by group

```
# subjectID must be factor. Site and sex should be numeric
#demo_combat_na_gam <- demo_combat_na
demo_combat_na_gam <- demo_combat_normed
demo_combat_na_gam$SUBJECTID %<>% as.factor
demo_combat_na_gam$site <- demo_combat_na_gam$site %>% gsub("T", "0",.) %>%
 ↪  gsub("P", "1",.) %>% as.numeric
demo_combat_na_gam$SEX <- demo_combat_na_gam$SEX %>% gsub("F", "0",.) %>%
 ↪  gsub("M", "1",.) %>% as.numeric
```

```
# per reviewer suggestion, run with fixed DoF by setting k and using fx =TRUE
# try two DoF because most estimated were between 1 and 3 and want to prevent
 ↪  overfitting on linear data
gamm_combat <- lapply(sc_names_normed, function(r) gam(formula = reformulate(
 ↪  c("s(AGE, by =SUBJECT_IDENTITY, bs =\"tp\", k= 3, fx =TRUE)", "gene_dosage",
 ↪  "SEX", "eTIVnormed", "site", "s(SUBJECTID, bs =\"re\", k= 3)"), response = r),
 ↪  data =demo_combat_na_gam, selection=TRUE, method = "REML", na.action=
 ↪  "na.omit"))

# name gamm list
names(gamm_combat) <- sc_names_normed

# add a GAMM for eTIVnormed that doesnt already contain it as a covariate
gamm_etiv <- gam(formula = reformulate( c("s(AGE, by =SUBJECT_IDENTITY, bs
 ↪  =\"tp\", k= 3, fx =TRUE)", "gene_dosage", "SEX", "site", "s(SUBJECTID, bs
 ↪  =\"re\", k= 3)"), response = "eTIVnormed"), data =demo_combat_na_gam,
 ↪  selection=TRUE, method = "REML", na.action= "na.omit")

# add eTIVnormed to gamm list
gamm_combat$eTIVnormed.combat.norm <-  gamm_etiv

# add eTIVnormed to list of names
all_names_normed <- c(sc_names_normed,"eTIVnormed.combat.norm")
```

same GAMM for restricted age range (<35)

```
gamm_combat_young <- lapply(sc_names_normed, function(r) gam(formula =
 ↪  reformulate( c("s(AGE, by =SUBJECT_IDENTITY, bs =\"tp\", k= 3, fx =TRUE)",
 ↪  "gene_dosage", "SEX", "eTIVnormed", "site", "s(SUBJECTID, bs =\"re\", k= 3)"),
 ↪  response = r), data =filter(demo_combat_na_gam, AGE <= 35), selection=TRUE,
 ↪  method = "REML", na.action= "na.omit"))

# name gamm list
names(gamm_combat_young) <- sc_names_normed

# add a GAMM for eTIVnormed that doesnt already contain it as a covariate
gamm_etiv_young <- gam(formula = reformulate( c("s(AGE, by =SUBJECT_IDENTITY, bs
 ↪  =\"tp\", k= 3, fx =TRUE)", "gene_dosage", "SEX", "site", "s(SUBJECTID, bs
 ↪  =\"re\", k= 3)"), response = "eTIVnormed"), data =filter(demo_combat_na_gam,
 ↪  AGE <= 35), selection=TRUE, method = "REML", na.action= "na.omit")

# add eTIVnormed to gamm list
gamm_combat_young$eTIVnormed.combat.norm <-  gamm_etiv_young
```

Gene dosage analysis from GAMMs

```
# get p-values for linear effect of gene dosage
gene_dosage_effect <- lapply(gamm_combat, function(l)
 ↪  summary(l)$p.table["gene_dosage", c("Estimate", "Pr(>|t|)")]) %>%
 ↪  do.call(rbind,.) %>% as.data.frame
colnames(gene_dosage_effect) <- c("gene_dosage_beta", "gene_dosage_p")
gene_dosage_effect$Region <- all_names_normed
```

```
# FDR correct
gene_dosage_effect$gene_dosage_fdr_q <- p.adjust(gene_dosage_effect$gene_dosage_p,
 ↪  method = "fdr")
gene_dosage_effect$fdr_sig <- gene_dosage_effect$gene_dosage_fdr_q < 0.05

# Bonferroni correct
gene_dosage_effect$gene_dosage_bonf_p <-
 ↪  p.adjust(gene_dosage_effect$gene_dosage_p, method = "bonferroni")
gene_dosage_effect$bonf_sig <- gene_dosage_effect$gene_dosage_bonf_p < 0.05

# mark significance
give_stars <- function(fdr, bonf){
  out <- ""
  if(fdr ==TRUE){
    out <- paste0(out,"*")
  }
  if(bonf==TRUE){
    out <- paste0(out,"*")
  }
  return(out)
}
gene_dosage_effect$sig <- NA
for(i in 1:nrow(gene_dosage_effect)){
  gene_dosage_effect[i,"sig"] <- give_stars(fdr =gene_dosage_effect[i,"fdr_sig"],
 ↪  bonf=gene_dosage_effect[i,"bonf_sig"])
}
```

make table for export with subregion gene dosage effects

```
# create data frame for export
save_dosage_effect <- filter(gene_dosage_effect, gene_dosage_fdr_q<0.05)[,
 ↪  c("gene_dosage_beta", "gene_dosage_p", "gene_dosage_fdr_q", "Region", "sig")]

# add whole amydgala and thalamus despite no FDR significance
save_dosage_effect <- rbind(save_dosage_effect,
 ↪  gene_dosage_effect[c("Thal_Whole_thalamus.combat.norm",
 ↪  "Amy_Whole_amygdala.combat.norm"), c("gene_dosage_beta", "gene_dosage_p",
 ↪  "gene_dosage_fdr_q", "Region", "sig")])
colnames(save_dosage_effect) <- c("beta", "p", "FDR q", "Region", "sig")

# get region names for matching
save_dosage_effect$Region <- gsub(".combat.norm", "", save_dosage_effect$Region)
save_dosage_effect$Region <- gsub("Thal_", "", save_dosage_effect$Region)
save_dosage_effect$Region <- gsub("Amy_", "", save_dosage_effect$Region)
save_dosage_effect$Region <- gsub("Hip_", "", save_dosage_effect$Region)

# merge with full names
#save_dosage_effect <- merge(x =save_dosage_effect, y =lut[c("Structure", "Name",
 ↪  "region_match")], by.x = "Region", by.y = "region_match", all.x =TRUE)
save_dosage_effect <- merge(x =save_dosage_effect, y =lut_unique[, c("Structure",
 ↪  "bilat_name", "bilat_match")], by.x = "Region", by.y = "bilat_match", all.x
 ↪  =TRUE)
```

```r
# edit structure label
save_dosage_effect$Structure <- paste(save_dosage_effect$Structure, "subregions")
 ↪  %>% gsub("whole brain subregions", "whole brain",.)

# add total ICV name
#save_dosage_effect[which(save_dosage_effect$Region== "eTIVnormed"),
 ↪  c("Structure", "bilat_name")] <-  c("whole brain", "total ICV")

# change structure label for whole thal, hip, amy
save_dosage_effect[which(save_dosage_effect$Region %in% c("Whole_amygdala",
 ↪  "Whole_thalamus", "Whole_hippocampus")),"Structure"] <-  "whole volumes"

# set order
save_dosage_effect$struct_order <- save_dosage_effect$Structure %>% gsub("whole
 ↪  brain",1,.) %>% gsub("whole volumes",2,.) %>% gsub("thalamus subregions",3,.)
 ↪  %>% gsub("hippocampus subregions",4,.) %>% gsub("amygdala subregions",5,.)

save_dosage_effect <- save_dosage_effect[with(save_dosage_effect,
 ↪  order(struct_order, beta)),]

#save_dosage_effect$Structure <- gsub("thalamus", "thal",
 ↪  save_dosage_effect$Structure)
#save_dosage_effect$Structure <- gsub("hippocampus", "hip",
 ↪  save_dosage_effect$Structure)
#save_dosage_effect$Structure <- gsub("amygdala", "amy",
 ↪  save_dosage_effect$Structure)
#save_dosage_effect$Region <- save_dosage_effect$Name
save_dosage_effect$Region <- save_dosage_effect$bilat_name
rownames(save_dosage_effect) <- NULL

# round
save_dosage_effect$beta %<>% round(., digits = 2) %>% sprintf("%.2f",.)
#save_dosage_effect$p %<>% round(., digits = 3) %>% sprintf("%.3f",.)
save_dosage_effect$p %<>% formatC(., format = "e", digits = 1)
#save_dosage_effect$`FDR q` %<>% round(., digits = 6) %>% sprintf("%.6f",.)
# g option formats as scientific only when saves space
save_dosage_effect$`FDR q` %<>% formatC(., format = "g", digits = 2)

# edit structure names
structure_dup <- duplicated(save_dosage_effect$Structure)
for(i in 1:length(structure_dup)){
  if(structure_dup[i]==TRUE){
    save_dosage_effect[i,"Structure"] <- ""
  }
}

save_dosage_effect_final <- save_dosage_effect[, c("Structure", "Region", "beta",
 ↪  "p", "FDR q", "sig")]

# export table
save_dosage_effect_out <- save_dosage_effect_final %>% gt() %>%
  tab_style(style = cell_text(weight = "bold"), locations =
   ↪  list(cells_column_labels())) %>%
```

```
  cols_align(align= "right", columns =everything())
save_dosage_effect_out
```

Structure	Region	beta	p	FDR q	sig
whole brain	total ICV	0.28	1.3e-03	0.0038	*
whole volumes	whole thalamus	-0.03	7.0e-01	0.8	
	whole amygdala	0.14	8.7e-02	0.17	
	whole hippocampus	0.47	7.3e-07	7.7e-06	**
thalamus subregions	mediodorsal	-0.36	3.3e-05	0.00015	**
	ventral lateral	-0.30	1.3e-04	0.00046	**
	lateral posterior	0.22	1.8e-02	0.047	*
	lateral geniculate	0.37	9.1e-05	0.00038	**
	medial ventral (reuniens)	0.39	1.1e-04	0.00042	**
hippocampus subregions	GC ML DG	0.41	1.4e-05	7.3e-05	**
	CA4	0.42	9.8e-06	5.9e-05	**
	subiculum	0.47	1.3e-07	1.9e-06	**
	CA1	0.48	3.9e-06	2.8e-05	**
	molecular layer	0.49	1.2e-06	1e-05	**
	hippocampal fissure	0.54	2.2e-08	4.7e-07	**
	hippocampal tail	0.61	1.4e-09	5.9e-08	**
amygdala subregions	accessory basal nucleus	0.21	1.9e-02	0.048	*
	paralaminar nucleus	0.28	1.8e-03	0.005	*
	basal nucleus	0.31	4.5e-04	0.0014	**

```
# gtsave(save_dosage_effect_out, filename = file.path(project,
 ↪  "figures/gene_dosage/gene_dosage_table.png"))
# gtsave(save_dosage_effect_out, filename = file.path(project,
 ↪  "figures/gene_dosage/gene_dosage_table.pdf"))
# gtsave(save_dosage_effect_out, filename = file.path(project,
 ↪  "figures/gene_dosage/gene_dosage_table.rtf"))
# gtsave(save_dosage_effect_out, filename = file.path(project,
 ↪  "figures/gene_dosage/gene_dosage_table.tex"))
```

plot gene dosage effects whole structure scatterplots

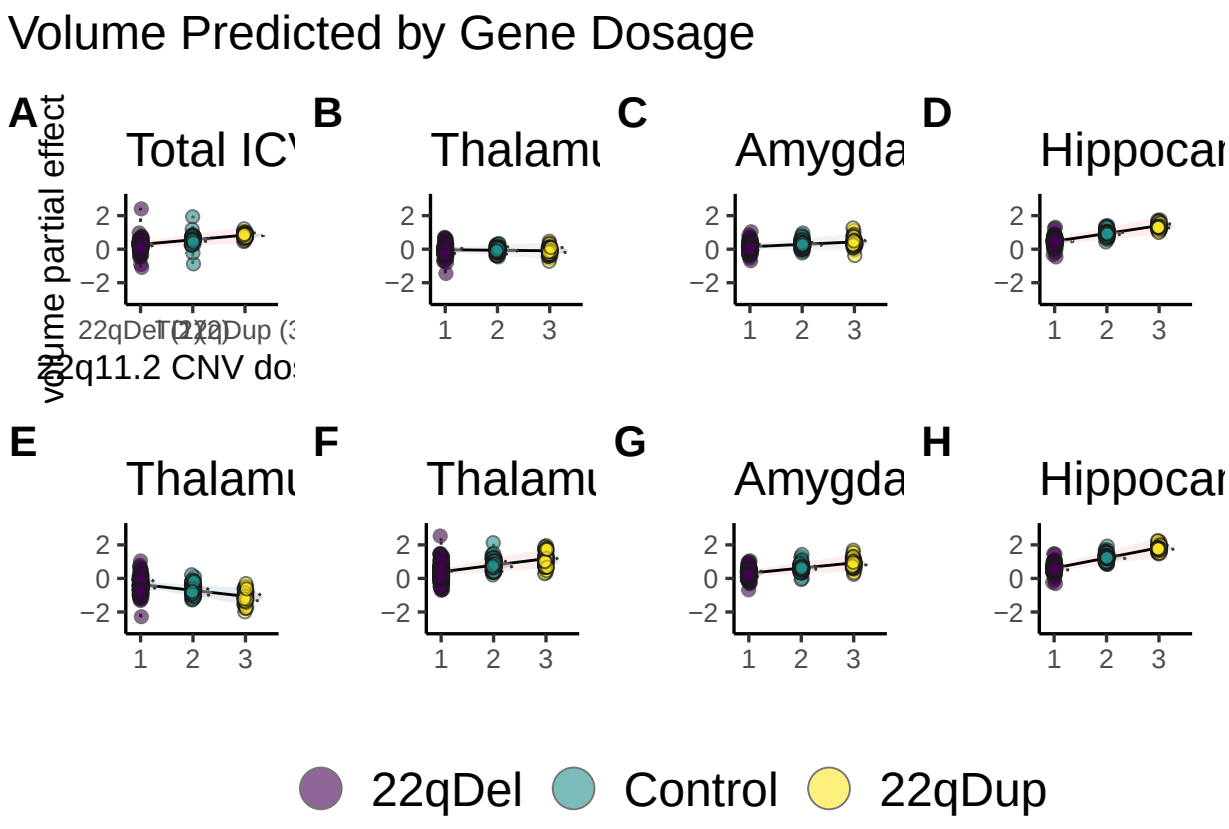
```
#  whole thal
#Thal_Whole <- effect(term= "gene_dosage", mod =lme4::lmer(formula =
 ↪  "Thal_Whole_thalamus.combat ~ gene_dosage + AGE + AGE2 + SEX + eTIVscaled +
 ↪  site + (1|SUBJECTID)", data =demo_combat_na, REML=TRUE)) %>% as.data.frame

# function to plot gene dosage scatterplot from gamm resids
gene_dosage_gamm_plot <- function(gam_list, name, title = "", xlab= "", ylab= "",
 ↪  xlabels = c(1,2,3), ribbon_fill = "lightgrey"){
  # get mgcViz object from gamm
  viz <- getViz(gam_list[[name]])
  # plot first parametric term (gene_dosage)
  pt <- pterm(viz, 1)
  # create plot object with fit line and 0.95 CI
  gt <- plot(pt)+ l_ciPoly(level = 0.95) + l_fitLine()
  # create ggplot
  plot <- ggplot() +
    # create confidence interval from estimate +/- 1.96 * the standard error in
     ↪  the mgcViz model/plot object
```

```
    geom_ribbon(data =gt$data$fit, inherit.aes = FALSE, aes(x = x, ymin=
     ↪  y-1.96*se, ymax = y+1.96*se), fill = ribbon_fill, alpha = 0.3)+
    # plot line from the mgcViz model fit estimate
    geom_line(data =gt$data$fit, inherit.aes = FALSE, aes(x = x, y = y), color =
     ↪  "black")+
    # plot kernel density estimate for residuals
    geom_half_violin(data =gt$data$res, inherit.aes = FALSE, aes(x = x, group= x,
     ↪  y = y), fill = "grey", color = "grey20", side = "r", lty = "dotted", alpha
     ↪  = 0.5)+
    # residual scatterplot from mgcViz res
    geom_point(data =gt$data$res, inherit.aes = FALSE, aes(x = x, y = y, fill =
     ↪  as.factor(x)), position=position_jitter(w= 0.03, h= 0, seed = 1), shape =
     ↪  21, color = "grey10", alpha = 0.6, size = 2)+
    # plot details
    scale_y_continuous(limits = c(-3,3), breaks = c(-2,0,2), minor_breaks = NULL)+
    scale_x_continuous(limits = c(0.85,3.5), breaks = c(1,2,3), minor_breaks =
     ↪  NULL, labels = xlabels)+
    scale_fill_viridis(discrete = TRUE, name = "", labels = c("22qDel", "Control",
     ↪  "22qDup"))+
    theme_classic(base_size = 13)+
    xlab(xlab)+
    ylab(ylab)+
    ggtitle(title)+
    guides(fill = guide_legend(override.aes = list(size = 7)))

  return(plot)
}

plot_ICV <- gene_dosage_gamm_plot(gam_list=gamm_combat, name =
 ↪  "eTIVnormed.combat.norm", title = "Total ICV", ribbon_fill = "rosybrown1",
 ↪  xlabels = c("22qDel (1)", "TD (2)", "22qDup (3)"), ylab= "volume partial
 ↪  effect", xlab= "22q11.2 CNV dosage")
plot_Thal_Whole <- gene_dosage_gamm_plot(gam_list=gamm_combat, name =
 ↪  "Thal_Whole_thalamus.combat.norm", title = "Thalamus", ribbon_fill =
 ↪  "lightgrey")
plot_Amy_Whole <- gene_dosage_gamm_plot(gam_list=gamm_combat, name =
 ↪  "Amy_Whole_amygdala.combat.norm", title = "Amygdala", ribbon_fill =
 ↪  "lightgrey")
plot_Hip_Whole <- gene_dosage_gamm_plot(gam_list=gamm_combat, name =
 ↪  "Hip_Whole_hippocampus.combat.norm", title = "Hippocampus", ribbon_fill =
 ↪  "rosybrown1")
plot_Thal_MD <- gene_dosage_gamm_plot(gam_list=gamm_combat, name =
 ↪  "Thal_MD_all.combat.norm", title = "Thalamus MD", ribbon_fill =
 ↪  "lightsteelblue")
plot_Thal_MV <- gene_dosage_gamm_plot(gam_list=gamm_combat, name =
 ↪  "Thal_MV_Re_.combat.norm", title = "Thalamus MV", ribbon_fill = "rosybrown1")
plot_Amy_basal <- gene_dosage_gamm_plot(gam_list=gamm_combat, name =
 ↪  "Amy_Basal_nucleus.combat.norm", title = "Amygdala Basal", ribbon_fill =
 ↪  "rosybrown1")
plot_Hip_tail <- gene_dosage_gamm_plot(gam_list=gamm_combat, name =
 ↪  "Hip_Hippocampal_tail.combat.norm", title = "Hippocampus Tail", ribbon_fill =
 ↪  "rosybrown1")
```

```
#ggsave(plot=plot_Thal_Whole, filename =file.path(project,
 ↪  "figures/gene_dosage/Thal_Whole.svg"), width= 5, height= 6, device = "svg")
#ggsave(plot=plot_Thal_Whole, filename =file.path(project,
 ↪  "figures/gene_dosage/Thal_Whole.pdf"), width= 5, height= 6, device = "pdf")
#ggsave(plot=plot_Thal_Whole, filename =file.path(project,
 ↪  "figures/gene_dosage/Thal_Whole.png"), width= 5, height= 6, device = "png",
 ↪  dpi= 300)
```

Create figure with all lm plots

```
# whole structure plots on top, selected subregions second row, panels labeled
 ↪  with bold letters
lm_fig_top <- (plot_ICV + plot_Thal_Whole + plot_Amy_Whole + plot_Hip_Whole)
 ↪  +plot_layout(ncol = 4, nrow = 1)
lm_fig_bottom <- (plot_Thal_MD + plot_Thal_MV + plot_Amy_basal + plot_Hip_tail)
 ↪  +plot_layout(ncol = 4, nrow = 1)

lm_fig_all <- lm_fig_top/lm_fig_bottom  + plot_layout(guides = "collect") +
 ↪  plot_annotation(tag_levels = 'A', title = "Volume Predicted by Gene Dosage") &
 ↪  theme(plot.tag = element_text(face = 'bold'), plot.title = element_text(size =
 ↪  18, hjust = 0)) + theme(legend.position = "bottom",
 ↪  legend.text=element_text(size = 18), legend.title =element_text(size = 18))

lm_fig_all
```




```
#ggsave(plot=lm_fig_all, filename =file.path(project,
 ↪  "figures/gene_dosage/gene_dosage_all.pdf"), width= 12, height= 8.5, device =
 ↪  "pdf")
```

```
#ggsave(plot=lm_fig_all, filename =file.path(project,
 ↪  "figures/gene_dosage/gene_dosage_all.png"), width= 12, height= 8.5, device =
 ↪  "png", dpi = 300)
```

Age curves from GAMMs

wrap all maturational analyses in a function so that they can be repeated with a truncated age subset of the input function allows for exact replication of age analyses with multiple age ranges

```
# the next few functions will be used inside of main_maturation()
# function to estimate smooths for plotting
smooth_estimates_se <- function(gamm, smooth, n){
  out <- smooth_estimates(gamm, smooth, n=n, partial_match = T)
  out$selo <- out$est - out$se
  out$sehi <- out$est + out$se
  return(out)
}

# function to take derivative output that includes an age smooth output age range
 ↪  where CI doesn't include zero
# adapted from
 ↪  https://github.com/pittnerdlab/22q11_longitudinal_cortical_sMRI/blob/main/01a_age_effects.R
get_sig_diff_ages_sc <- function(gam, smooth, group1= "CONTROL", group2){
  # get derivative of specified smooth from gam
  diff <- difference_smooths(model =gam, smooth=smooth, n= 1000)
  # filter for only chosen groups
  gdiff <- filter(diff, level_1==group1 & level_2==group2)
  # get points where confidence interval doesn't include zero
  sig <- sign(gdiff$lower) == sign(gdiff$upper)
  # get list of ages
  agelist <- gdiff$AGE[sig]
  ## create age range from list of ages
  # set age gap (years) between significant ages to be considered new range
  sigjump_brain<-0.23
  j= 1
  ranges = ""
  if(length(agelist)>0) {
    ranges<-round(agelist[[j]], digits = 1)
  }
  while (j < length(agelist)) {
    j<-j+1
    gdiff<-agelist[[j]]-agelist[[j-1]]
    if (gdiff > sigjump_brain) {
      ranges<-paste0(ranges,"-", round(agelist[[j-1]], digits = 1),"|",
 ↪  round(agelist[[j]], digits = 1))
    }
    if(j==length(agelist)){
      ranges<-paste0(ranges,"-", round(agelist[[j]], digits = 1))
    }
  }
  return(ranges)
}
```

```r
# giant function to run all age curve analyses and output a set of stats and plots
 ↪  as a list object
main_maturation <- function(gam_list){
  # create empty list object to fill with results
  out <- list()

  # add initial gam list
  out$gam_list <- gam_list

  # get smooth tables from gam
  stables <- lapply(gam_list, function(g) summary(g, freq=T)$s.table)
  # retun smooth tables
  out$stables <- stables

  # get p-vals
  del_age_pvals <- lapply(stables,
 ↪  function(s)s["s(AGE):SUBJECT_IDENTITYPATIENT-DEL", c("F", "p-value")]) %>%
 ↪  do.call(rbind,.) %>% data.frame %>% rename("age_p_val"= "p.value")
  del_age_pvals$Region <- rownames(del_age_pvals)
  del_age_pvals$SUBJECT_IDENTITY <- "PATIENT-DEL"

  dup_age_pvals <- lapply(stables,
 ↪  function(s)s["s(AGE):SUBJECT_IDENTITYPATIENT-DUP", c("F", "p-value")]) %>%
 ↪  do.call(rbind,.) %>% data.frame %>% rename("age_p_val"= "p.value")
  dup_age_pvals$Region <- rownames(dup_age_pvals)
  dup_age_pvals$SUBJECT_IDENTITY <- "PATIENT-DUP"

  hcs_age_pvals <- lapply(stables, function(s)s["s(AGE):SUBJECT_IDENTITYCONTROL",
 ↪  c("F", "p-value")]) %>% do.call(rbind,.) %>% data.frame  %>%
 ↪  rename("age_p_val"= "p.value")
  hcs_age_pvals$Region <- rownames(hcs_age_pvals)
  hcs_age_pvals$SUBJECT_IDENTITY <- "CONTROL"

  # correct for multiple comparisons (one test for age effects per group per
   ↪  network)
  all_age_pvals <- rbind(del_age_pvals, dup_age_pvals, hcs_age_pvals)
  all_age_pvals$age_p_val_fdr <- all_age_pvals$age_p_val %>% p.adjust(., method =
 ↪  "fdr")
  all_age_pvals

  # make pretty table for export
  out_age_pvals <- all_age_pvals[, c("SUBJECT_IDENTITY", "Region", "F",
 ↪  "age_p_val", "age_p_val_fdr")]
  rownames(out_age_pvals) <- NULL
  out_age_pvals <- rename(out_age_pvals, "p"= "age_p_val", "FDR_q"=
 ↪  "age_p_val_fdr", "Group"= "SUBJECT_IDENTITY")
  out_age_pvals$Group <- out_age_pvals$Group %>% gsub("CONTROL", "TD",.) %>%
 ↪  gsub("PATIENT-DEL", "22qDel",.) %>% gsub("PATIENT-DUP", "22qDup",.)
  #out_age_pvals$Region <- gsub(".combat", "", out_age_pvals$Region)
  out_age_pvals$Region <- gsub(".combat.norm", "", out_age_pvals$Region)
  out_age_pvals$F %<>% round(., digits = 2) %>% sprintf("%.4f",.)
  out_age_pvals$p %<>% signif(., digits = 2) %>% sprintf("%.3f",.)
```

```
 out_age_pvals$FDR_q %<>% signif(., digits = 3) %>% sprintf("%.4f",.)
 # retun age p-vals
 out$out_age_pvals <- out_age_pvals

 # get only significant results
 out_age_pvals_sig <- filter(out_age_pvals, FDR_q<0.05)
 out_age_pvals_sig

 # get smooth estimate
 smooth_all_combat <- lapply(gam_list, function(g) smooth_estimates_se(gamm=g,
↪  smooth= "s(AGE)", n= 1000))
 #names(smooth_all_combat) <- sc_names_combat
 names(smooth_all_combat) <- all_names_normed
 # return smooth estimate
 out$smooth_all_combat <- smooth_all_combat

 #get difference with TD smooth
 # get all regions with a significant effect
 age_regions <- out_age_pvals_sig$Region %>% unique %>% sort
 age_names <- paste0(age_regions,".combat.norm")

 # get age ranges where del or dup significantly differ from td
 agediff_td_del <- lapply(gam_list[age_names], function(g)
↪  get_sig_diff_ages_sc(gam=g, smooth= "s(AGE)", group1= "CONTROL", group2=
↪  "PATIENT-DEL"))
 agediff_td_dup <- lapply(gam_list[age_names], function(g)
↪  get_sig_diff_ages_sc(gam=g, smooth= "s(AGE)", group1= "CONTROL", group2=
↪  "PATIENT-DUP"))

 # organize age group differences by region and group
 age_group_diffs <- data.frame(FullName =names(agediff_td_del), td_del =
↪  as.vector(unlist(agediff_td_del)), td_dup= as.vector(unlist(agediff_td_dup)))
 age_group_diffs$Region <- gsub(".combat.norm", "", age_group_diffs$FullName)
 # return age group diffs
 out$age_group_diffs <- age_group_diffs

 # convert to true/false for regions with or without a difference
 age_group_tf <- (age_group_diffs %>% as.matrix %>% nchar) != 0
 age_group_tf <- data.frame(age_group_tf)
 age_group_tf$Region <- age_group_diffs$Region
 age_group_tf$either <- age_group_tf$td_del == TRUE | age_group_tf$td_dup == TRUE

 #make table of age differences for export
 age_group_export <- age_group_diffs
 age_group_export$bilat_match <- age_group_export$Region %>% gsub("Thal_", "",.)
↪  %>% gsub("Hip_", "",.) %>% gsub("Amy_", "",.)
 age_group_export  <- merge(x = age_group_export, y =lut_unique, by =
↪  "bilat_match")

 age_group_export_final <- age_group_export[, c("Structure", "bilat_name",
↪  "td_del", "td_dup")]
 colnames(age_group_export_final) <- c("Structure", "Region", "diff_TD_22qDel",
  ↪  "diff_TD_22qDup")
```

```r
 age_group_export_final$struct_order <- age_group_export_final$Structure %>%
↪  gsub("whole brain",1,.) %>% gsub("thalamus",2,.) %>% gsub("hippocampus",3,.)
↪  %>% gsub("amygdala",4,.)
 age_group_export_final <- age_group_export_final[with(age_group_export_final,
↪  order(struct_order,Region)),]
 age_group_export_final <- subset(age_group_export_final, select=-struct_order)

 # edit structure names
 structure_dup <- duplicated(age_group_export_final$Structure)
 for(i in 1:length(structure_dup)){
   if(structure_dup[i]==TRUE){
     age_group_export_final[i,"Structure"] <- ""
   }
 }

 # return age group final table
 out$age_group_export_final <- age_group_export_final
 #write.csv(age_group_export_final[, c("Structure", "Region", "diff_TD_22qDel",
  ↪  "diff_TD_22qDup")], file =file.path(project,
  ↪  "figures/age/age_differences.csv"), row.names = FALSE)

 #plot summary of GAMM results
 # split data frames by group
 sig_del <- filter(out_age_pvals_sig, Group== "22qDel")
 sig_dup <- filter(out_age_pvals_sig, Group== "22qDup")
 sig_hcs <- filter(out_age_pvals_sig, Group== "TD")

 # list of unique regions with significance in 22q or TD
 sig_regions_compare <- data.frame(Region=sort(unique(out_age_pvals_sig$Region)))
 row.names(sig_regions_compare) <- sig_regions_compare$Region

 for (region in sig_regions_compare$Region){
   sig_regions_compare[region,"22qDel"] <- region %in% sig_del$Region
   #sig_regions_compare[region,"Del_u30"] <- region %in% sig_del_u30$Region
   sig_regions_compare[region,"22qDup"] <- region %in% sig_dup$Region
   #sig_regions_compare[region,"Dup_u30"] <- region %in% sig_dup_u30$Region
   sig_regions_compare[region,"TD"]<- region %in% sig_hcs$Region
   #sig_regions_compare[region,"TD_u30"] <- region %in% sig_hcs_u30$Region
 }
 #sig_regions_compare

 # create new column with TRUE if not also significant in TD
 sig_regions_compare$TD_sig <- NA
 for (r in 1:nrow(sig_regions_compare)){
   td <- sig_regions_compare[r,"TD"]
   #td30 <- sig_regions_compare[r,"TD_u30"]
   if (td ==TRUE){
     sig_regions_compare[r,"TD_sig"] <- TRUE
   }else{
     sig_regions_compare[r,"TD_sig"] <- FALSE
   }
 }
```

```
# return comparison of regions
out$sig_regions_compare <- sig_regions_compare

# create long df for plotting
setDT(sig_regions_compare)
idvars = c("Region", "TD_sig")
compare_long <- melt.data.table(sig_regions_compare, id.vars = idvars,
↪  measure.vars = names(sig_regions_compare)[which(!names(sig_regions_compare)
↪  %in% idvars)])
compare_long$Region %<>% as.factor

# order by region, then group
setorder(compare_long, Region, variable)

# for plotting, create new column with TRUE if the value at row r is equal to
 ↪  row r-1 (excluding rows where variable is del_all)
compare_long$postmatch <- NA
for (r in 1:(nrow(compare_long)-1)){
  val <- compare_long[r,"value"]
  postval <- compare_long[(r+1),"value"]
  if (val ==TRUE & postval ==TRUE){
    compare_long[r,"postmatch"] <- TRUE
  }else{
    compare_long[r,"postmatch"] <- FALSE
  }
}
# set to NA group that will be rightmost column of plot
groups <- levels(compare_long$variable)
last_group <- groups[length(groups)]
compare_long[which(compare_long$variable ==last_group),"postmatch"] <- NA

# create column with TRUE if pt group is significant but TD are not
compare_long$pt_sig_only <- NA
for (r in 1:(nrow(compare_long))){
  if(compare_long[r,"TD_sig"]==FALSE & compare_long[r,"value"]==TRUE){
    #if(compare_long[r,"TD"]==FALSE & compare_long[r,"value"]==TRUE){
    compare_long[r,"pt_sig_only"] <- TRUE
  }
  #if(compare_long[r,"variable"]== "TD_all" | compare_long[r,"variable"]==
   ↪  "TD_u30"){
  if(compare_long[r,"variable"]== "TD_all"){
    compare_long[r,"pt_sig_only"] <- NA
  }
}
# replace NA with FALSE
compare_long <- replace_na(compare_long, list(pt_sig_only =FALSE))

# edit region names to remove trailing underscore and "_all", replace other
 ↪  underscores with space, and capitalize first letter without editing
 ↪  subsequent letters
#compare_long$Region_edit <- compare_long$Region %>% gsub("_$", "",.) %>%
 ↪  gsub("_all", "",.) %>% gsub("_", " ",.) %>% gsub("\\b([a-z])", "\\U\\1",.,
 ↪  perl =TRUE)
```

```
 compare_long$bilat_match <- compare_long$Region %>% gsub("Thal_", "",.) %>%
↪  gsub("Hip_", "",.) %>% gsub("Amy_", "",.)
 compare_long <- merge(x = compare_long, y =lut_unique[, c("bilat_name",
↪  "bilat_match", "Structure")], by = "bilat_match", all.x =TRUE)
 #compare_long$Region_edit <- paste(compare_long$Structure,
  ↪  compare_long$bilat_name)
 compare_long$Region_edit1 <- compare_long$bilat_name %>% gsub("hippocampal
↪  amygdala transition area", "HATA",.) %>% gsub("hippocampus", "",.) %>%
↪  gsub("thalamus", "",.) %>%   gsub("amygdala", "",.) %>% gsub(" $", "",.) %>%
↪  gsub("  ", " ",.)

 region_edit_df <- data.frame(col1= compare_long$Region_edit1, col2=
↪  compare_long$Structure)
 region_edit_df <- region_edit_df[!duplicated(region_edit_df),]
 setorder(region_edit_df, col2, col1)
 region_levels<- paste(region_edit_df$col1, region_edit_df$col2)

 compare_long$Region_edit <- factor(x =paste(compare_long$Region_edit1,
↪  compare_long$Structure), levels = region_levels)
 #compare_long$Region_edit <- paste(compare_long$Structure,
  ↪  compare_long$Region_edit1)
 #compare_long$Region_edit <- compare_long$Region_edit %>% gsub("hippocampal
  ↪  transition area hippocampus", "HATA hippocampus",.) %>% gsub("  ", " ",.)

 # add column with true if there is a group difference between patients and
  ↪  controls
 #compare_long <- merge(x = compare_long, y = age_group_tf[, c("Region",
  ↪  "either")], by = "Region")
 compare_long$gdiff <- NA
 for (i in 1:nrow(compare_long)){
   #print(i)
   region <- compare_long[i,"Region"] %>% as.matrix %>% as.character
   #print(region)
   variable <- compare_long[i,"variable"] %>% as.matrix %>% as.character
   #print(variable)
   gdiff <- NA
   if(variable == "22qDel"){
     gdiff <- filter(age_group_tf, Region== region)$td_del
   }else if(variable == "22qDup"){
     gdiff <- filter(age_group_tf, Region== region)$td_dup
   }else if(variable == "TD"){
     gdiff <- filter(age_group_tf, Region== region)$either
   }
   #print(gdiff)
   compare_long[i,"gdiff"] <- gdiff
 }

 # return compare_long
 out$compare_long <- compare_long

 # plot
 gamm_compare <- ggplot(data = compare_long, aes(y =Region_edit, x =variable))+
   geom_line(aes(group=Region, color =postmatch), alpha = 0.4, show.legend =
    ↪  FALSE)+
```

```
    scale_color_manual(values = c("white", "black"))+
    new_scale_color()+
    geom_point(aes(shape =value, size =gdiff, color =pt_sig_only, alpha =value))+
    scale_shape_manual(values = c(1,16))+
    scale_alpha_manual(values = c(0.3,1))+
    scale_size_manual(values = c(2,4))+
    scale_color_manual(values = c("black", "red"), na.value = "black")+
    labs(shape = "Age: FDR q < 0.05", alpha = "Age: FDR q < 0.05", size = "Group
     ↪  Difference", color = "Significant in\nPatients only")+
    guides(alpha = guide_legend(order = 1),
           shape = guide_legend(order = 1, override.aes =list(size = 2)),
           #shape = guide_legend(order = 1, override.aes =list(size = 4, shape =
            ↪  c(0,15))),
           size  = guide_legend(order = 2),
           #color  = guide_legend(order = 3, override.aes =list(size = 4, shape =
            ↪  15)))+
           color  = guide_legend(order = 3, override.aes =list(size = 2)))+
    scale_x_discrete(position = "top")+
    xlab(NULL)+
    ylab("Region")+
    ggtitle("Significant Age Effects by Cohort")+
    theme_bw(base_size = 12)+
    theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
     ↪  axis.ticks.x =element_blank())

  #gamm_compare
  # return gamm comparison plot
  out$gamm_compare <- gamm_compare
  #ggsave(plot=gamm_compare, filename =file.path(project,
   ↪  "figures/age/age_gamm_group_compare.pdf"), width= 7, height= 5, device =
   ↪  "pdf")
  #ggsave(plot=gamm_compare, filename =file.path(project,
   ↪  "figures/age/age_gamm_group_compare.png"), width= 7, height= 5, device =
   ↪  "png", dpi = 300)

  #return final list object
  return(out)
}
```

do maturation analyses

```
# full age range
maturation_full <- main_maturation(gam_list=gamm_combat)
# removing oldest participants (over 35 years)
maturation_young <- main_maturation(gam_list=gamm_combat_young)
```

in smooth tables, mark age ranges of significant difference to controls

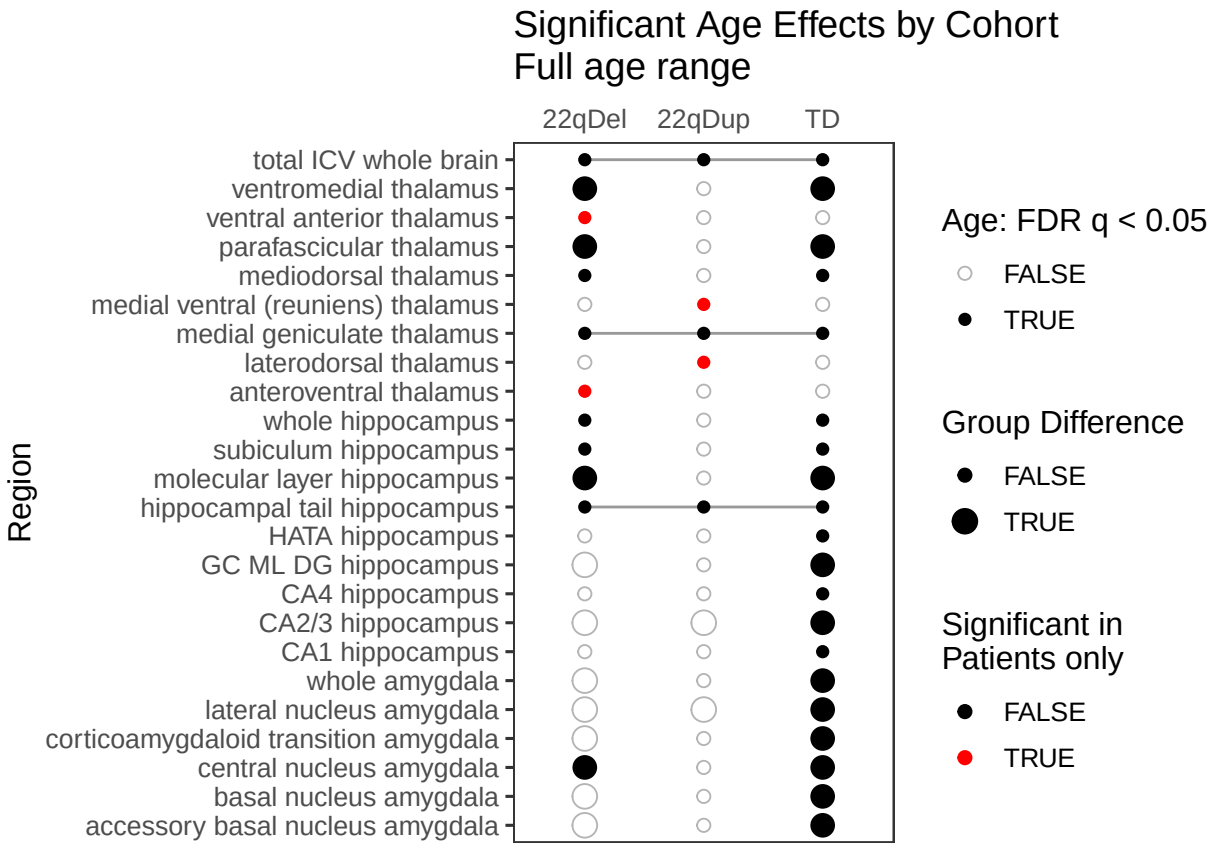
```
# function to add true/false for significant difference to TD group to smooth
 ↪  tables
gdiff_smooth <- function(mature){
  # make new list of smooths to edit
  mature$smooth_all_gdiff <- mature$smooth_all_combat
  # create a column to mark group difference from TD, and first set all to FALSE
```

```
for(i in 1:length(mature$smooth_all_gdiff)){
  mature$smooth_all_gdiff[[i]]$gdiff <- FALSE
}
# list of regions with some group diff
regions <- mature$age_group_diffs$FullName
# update smooth_all_gdiff
for(r in 1:length(regions)){
  name <- mature$age_group_diffs[r,"FullName"]
  td_del <- mature$age_group_diffs[r,"td_del"]
  td_dup <- mature$age_group_diffs[r,"td_dup"]
  # get periods of significant difference
  if(nchar(td_del)>1){
    # split individual periods
    periods <- str_split(string=td_del, pattern= "\\|")[[1]]
    for(t in 1:length(periods)){
      # get start and stop age
      ages <- str_split(string=periods[t], pattern= "-")[[1]]
      start <- ages[1] %>% as.numeric
      end <- ages[2] %>% as.numeric
      # update smooth_all_gdiff
      for(a in 1:nrow(mature$smooth_all_gdiff[[name]])){
        age <- mature$smooth_all_gdiff[[name]][a,"AGE"] %>% as.numeric
        group <- mature$smooth_all_gdiff[[name]][a,"SUBJECT_IDENTITY"]
        if(age >= start & age <= end & group == "PATIENT-DEL"){
          mature$smooth_all_gdiff[[name]][a,"gdiff"] <- TRUE
        }
      }
    }
  }
  if(nchar(td_dup)>1){
    # split individual periods
    periods <- str_split(string=td_dup, pattern= "\\|")[[1]]
    for(t in 1:length(periods)){
      # get start and stop age
      ages <- str_split(string=periods[t], pattern= "-")[[1]]
      start <- ages[1] %>% as.numeric
      end <- ages[2] %>% as.numeric
      # update smooth_all_gdiff
      for(a in 1:nrow(mature$smooth_all_gdiff[[name]])){
        age <- mature$smooth_all_gdiff[[name]][a,"AGE"] %>% as.numeric
        group <- mature$smooth_all_gdiff[[name]][a,"SUBJECT_IDENTITY"]
        if(age >= start & age <= end & group == "PATIENT-DUP"){
          mature$smooth_all_gdiff[[name]][a,"gdiff"] <- TRUE
        }
      }
    }
  }
}
# make smooth tables with non gdiff estimates set to zero
mature$smooth_sig_gdiff <- mature$smooth_all_gdiff
for(i in 1:length(mature$smooth_sig_gdiff)){
  for(a in 1:nrow(mature$smooth_sig_gdiff[[i]])){
    tf <- mature$smooth_sig_gdiff[[i]][a,"gdiff"]
```

```
      if(tf==FALSE){
        mature$smooth_sig_gdiff[[i]][a,"est"] <- NA
      }
    }
  }
  return(mature)
}

# update smooth tables with group difference indicator
mat_full_gdiff <- gdiff_smooth(maturation_full)
mat_young_gdiff <- gdiff_smooth(maturation_young)

#mat_full_gdiff$smooth_all_gdiff[["Hip_CA3.combat.norm"]]$gdiff %>% sum
```

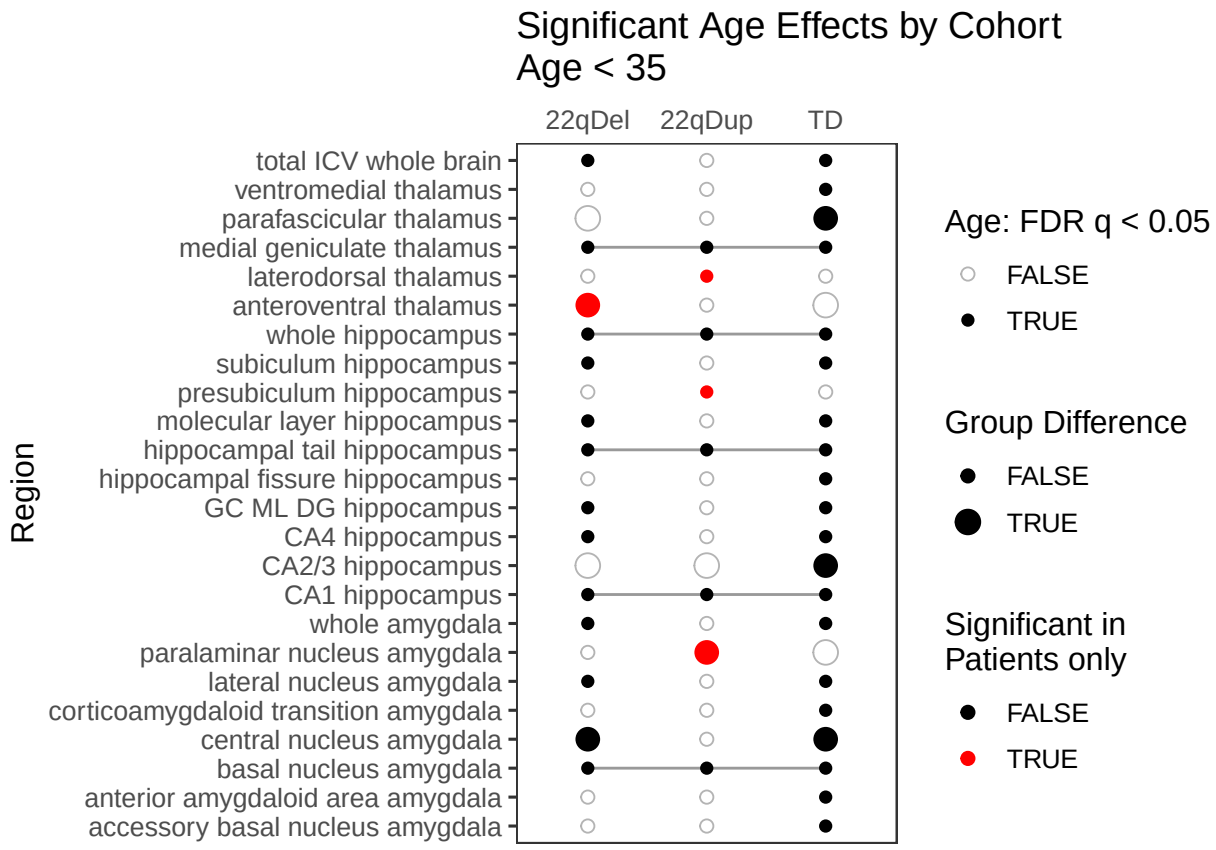
save GAMM comparison plots

```
# full age range
maturation_full$gamm_compare+ggtitle("Significant Age Effects by Cohort\nFull age
 ↪  range")
```




```
#ggsave(plot=maturation_full$gamm_compare+ggtitle("Significant Age Effects by
 ↪  Cohort\nFull age range"), filename =file.path(project,
 ↪  "figures/age/age_gamm_group_compare_full.pdf"), width= 7, height= 5, device =
 ↪  "pdf")
#ggsave(plot=maturation_full$gamm_compare+ggtitle("Significant Age Effects by
 ↪  Cohort\nFull age range"), filename =file.path(project,
 ↪  "figures/age/age_gamm_group_compare_full.png"), width= 7, height= 5, device =
 ↪  "png", dpi = 300)
```

```
# younger age range
maturation_young$gamm_compare+ggtitle("Significant Age Effects by Cohort\nAge <
 ↪  35")
```




```
#ggsave(plot=maturation_young$gamm_compare+ggtitle("Significant Age Effects by
 ↪  Cohort\nAge < 35"), filename =file.path(project,
 ↪  "figures/age/age_gamm_group_compare_u35.pdf"), width= 7, height= 5, device =
 ↪  "pdf")
#ggsave(plot=maturation_young$gamm_compare+ggtitle("Significant Age Effects by
 ↪  Cohort\nAge < 35"), filename =file.path(project,
 ↪  "figures/age/age_gamm_group_compare_u35.png"), width= 7, height= 5, device =
 ↪  "png", dpi = 300)
```

plot chosen GAMMS

```
# function to plot all age smooths with partial residuals
plot_gamm_resid <- function(list, name, xlab= "", ylab= "", ylim= c(-1,1), xlim=
 ↪  c(5.9,23), title = ""){
  gam_list <- list$gam_list
  smooth_all_combat <- list$smooth_all_combat
  if(title == ""){
    plot_title <- paste(gsub("_", " ", gsub(".combat.norm", "", name)),"Volume")
  }else{
    plot_title <- title
  }
  #resid <- add_partial_residuals(data =demo_combat_na, model =gam_list[[name]])
```

```
 resid <- add_partial_residuals(data =gam_list[[name]]$model, model
↪  =gam_list[[name]])
 resid_del <- filter(resid, SUBJECT_IDENTITY== "PATIENT-DEL")
 resid_del$yvar <- resid_del$`s(AGE):SUBJECT_IDENTITYPATIENT-DEL`
 resid_dup <- filter(resid, SUBJECT_IDENTITY== "PATIENT-DUP")
 resid_dup$yvar <- resid_dup$`s(AGE):SUBJECT_IDENTITYPATIENT-DUP`
 resid_hcs <- filter(resid, SUBJECT_IDENTITY== "CONTROL")
 resid_hcs$yvar <- resid_hcs$`s(AGE):SUBJECT_IDENTITYCONTROL`
 ggplot()+
   geom_point(data = resid_del, aes(x =AGE, y = yvar), alpha = 0.8, shape = 21,
    ↪  color = "gray20", fill =viridis(3)[1]) +
   geom_line(data = resid_del, aes(x =AGE, y = yvar, group=SUBJECTID), alpha =
    ↪  0.6, shape = 21, color =viridis(3)[1])+
   geom_point(data = resid_hcs, aes(x =AGE, y = yvar), alpha = 0.8, shape = 21,
    ↪  color = "gray20", fill =viridis(3)[2]) +
   geom_line(data = resid_hcs, aes(x =AGE, y = yvar, group=SUBJECTID), alpha =
    ↪  0.6, shape = 21, color =viridis(3)[2])+
   geom_point(data = resid_dup, aes(x =AGE, y = yvar), alpha = 0.8, shape = 21,
    ↪  color = "gray20", fill =viridis(3)[3]) +
   geom_line(data = resid_dup, aes(x =AGE, y = yvar, group=SUBJECTID), alpha =
    ↪  0.8, shape = 21, color = "orange")+
   #scale_shape_manual(values = c(17, 16))+
   # td
   geom_ribbon(data = filter(smooth_all_combat[[name]], SUBJECT_IDENTITY==
    ↪  "CONTROL"),
               aes(x = AGE, ymin = selo, ymax = sehi, fill = SUBJECT_IDENTITY),
                ↪  alpha = .3, linetype = 0)+
   geom_line(data = filter(smooth_all_combat[[name]], SUBJECT_IDENTITY==
    ↪  "CONTROL"),
             aes(x = AGE, y = est, color = SUBJECT_IDENTITY), size =
              ↪  1)+theme_bw()+
   # dup
   geom_ribbon(data = filter(smooth_all_combat[[name]], SUBJECT_IDENTITY==
    ↪  "PATIENT-DUP"),
               aes(x = AGE, ymin = selo, ymax = sehi, fill = SUBJECT_IDENTITY),
                ↪  alpha = .6, linetype = 0)+
   geom_line(data = filter(smooth_all_combat[[name]], SUBJECT_IDENTITY==
    ↪  "PATIENT-DUP"),
             aes(x = AGE, y = est, color = SUBJECT_IDENTITY), size =
              ↪  1)+theme_bw()+
   # del
   geom_ribbon(data = filter(smooth_all_combat[[name]], SUBJECT_IDENTITY==
    ↪  "PATIENT-DEL"),
               aes(x = AGE, ymin = selo, ymax = sehi, fill = SUBJECT_IDENTITY),
                ↪  alpha = .3, linetype = 0)+
   geom_line(data = filter(smooth_all_combat[[name]], SUBJECT_IDENTITY==
    ↪  "PATIENT-DEL"),
             aes(x = AGE, y = est, color = SUBJECT_IDENTITY), size =
              ↪  1)+theme_bw()+
   #scale_fill_manual(values = c("CONTROL" = viridis(3)[2], "PATIENT-DEL" =
    ↪  viridis(3)[1], "PATIENT-DUP" = "black"), labels = c("Control", "22qDel",
    ↪  "22qDup")) +
   scale_fill_manual(values = c("CONTROL" = viridis(3)[2], "PATIENT-DEL" =
    ↪  viridis(3)[1], "PATIENT-DUP" =viridis(3)[3]), labels = c("Control",
    ↪  "22qDel", "22qDup")) +
```

```
    scale_color_manual(values = c("CONTROL" = viridis(3)[2], "PATIENT-DEL" =
     ↪  viridis(3)[1], "PATIENT-DUP" = "orange"), labels = c("Control", "22qDel",
     ↪  "22qDup")) +
    #scale_x_continuous(limits = xlim, expand = c(0,0))+
    #scale_y_continuous(limits = ylim, expand = c(0,0))+
    theme_classic() +
    theme(legend.title = element_blank())+
    theme(axis.title.y = element_text(angle = 0, vjust= 0.5))+
    ylab(ylab)+
    xlab(xlab)+
    #ggtitle(paste(gsub("_", " ", gsub(".combat", "", name)),"Volume"))
    ggtitle(plot_title)
}
```

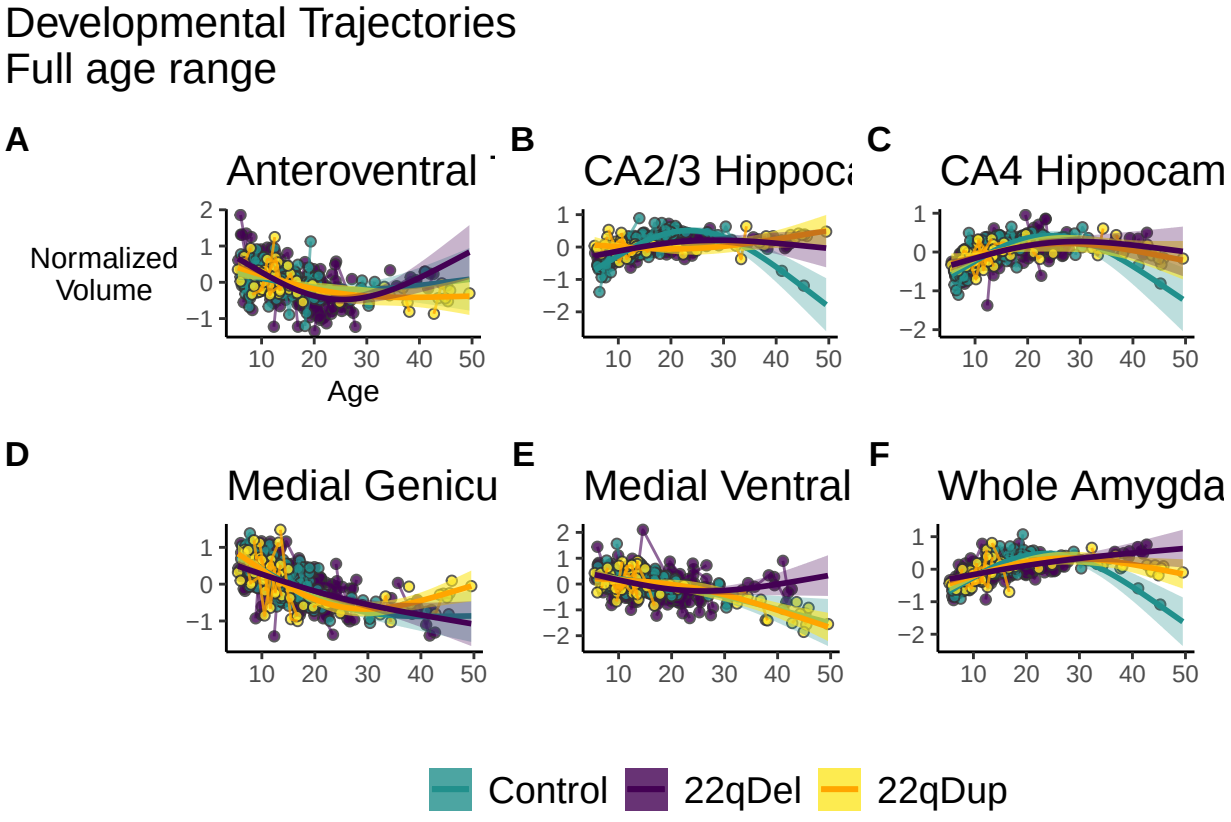
```
# function to plot all age smooths with age ranges of significant difference from
 ↪  controls highlighted
plot_gamm_gdiff <- function(list, name, xlab= "", ylab= "", ylim= c(-1,1), xlim=
 ↪  c(5.9,23), title = ""){
  gam_list <- list$gam_list
  smooth_all_gdiff <- list$smooth_all_gdiff
  smooth_sig_gdiff <- list$smooth_sig_gdiff
  if(title == ""){
    plot_title <- paste(gsub("_", " ", gsub(".combat.norm", "", name)),"Volume")
  }else{
    plot_title <- title
  }
  ggplot()+
    # td
    geom_ribbon(data = filter(smooth_all_gdiff[[name]], SUBJECT_IDENTITY==
     ↪  "CONTROL"),
                aes(x = AGE, ymin = selo, ymax = sehi, fill = SUBJECT_IDENTITY),
                 ↪  alpha = .3, linetype = 0)+
    geom_line(data = filter(smooth_all_gdiff[[name]], SUBJECT_IDENTITY==
     ↪  "CONTROL"),
              aes(x = AGE, y = est, color = SUBJECT_IDENTITY), size =
               ↪  0.5)+theme_bw()+
    # dup
    geom_ribbon(data = filter(smooth_all_gdiff[[name]], SUBJECT_IDENTITY==
     ↪  "PATIENT-DUP"),
                aes(x = AGE, ymin = selo, ymax = sehi, fill = SUBJECT_IDENTITY),
                 ↪  alpha = .6, linetype = 0)+
    geom_line(data = filter(smooth_all_gdiff[[name]], SUBJECT_IDENTITY==
     ↪  "PATIENT-DUP"),
              aes(x = AGE, y = est, color = SUBJECT_IDENTITY), size =
               ↪  0.5)+theme_bw()+
    geom_line(data = filter(smooth_sig_gdiff[[name]], SUBJECT_IDENTITY==
     ↪  "PATIENT-DUP"),
              aes(x = AGE, y = est, color = SUBJECT_IDENTITY), size =
               ↪  1.5)+theme_bw()+
    # geom_point(data = filter(smooth_all_gdiff[[name]], SUBJECT_IDENTITY==
     ↪  "PATIENT-DUP" & gdiff == TRUE),
    #           aes(x = AGE, y = est), color = "orange", size = 1, alpha =
     ↪  1)+theme_bw()+
```

```
    # del
    geom_ribbon(data = filter(smooth_all_gdiff[[name]], SUBJECT_IDENTITY==
     ↪  "PATIENT-DEL"),
                aes(x = AGE, ymin = selo, ymax = sehi, fill = SUBJECT_IDENTITY),
                 ↪  alpha = .3, linetype = 0)+
    geom_line(data = filter(smooth_all_gdiff[[name]], SUBJECT_IDENTITY==
     ↪  "PATIENT-DEL"),
              aes(x = AGE, y = est, color = SUBJECT_IDENTITY), size = 0.5, na.rm =
               ↪  TRUE)+theme_bw()+
    geom_line(data = filter(smooth_sig_gdiff[[name]], SUBJECT_IDENTITY==
     ↪  "PATIENT-DEL"),
              aes(x = AGE, y = est, color = SUBJECT_IDENTITY), size = 1.5, na.rm =
               ↪  TRUE)+theme_bw()+
    # geom_point(data = filter(smooth_all_gdiff[[name]], SUBJECT_IDENTITY==
     ↪  "PATIENT-DEL" & gdiff == TRUE),
    #           aes(x = AGE, y = est, color =SUBJECT_IDENTITY), shape = 19, color
     ↪  = "blue", size = 1, alpha = 1)+theme_bw()+
    #scale_fill_manual(values = c("CONTROL" = viridis(3)[2], "PATIENT-DEL" =
     ↪  viridis(3)[1], "PATIENT-DUP" = "black"), labels = c("Control", "22qDel",
     ↪  "22qDup")) +
    scale_fill_manual(values = c("CONTROL" = viridis(3)[2], "PATIENT-DEL" =
     ↪  viridis(3)[1], "PATIENT-DUP" =viridis(3)[3]), labels = c("Control",
     ↪  "22qDel", "22qDup")) +
    #scale_color_manual(values = c("CONTROL" = viridis(3)[2], "PATIENT-DEL" =
     ↪  viridis(3)[1], "PATIENT-DUP" = viridis(3)[3]), labels = c("Control",
     ↪  "22qDel", "22qDup")) +
    scale_color_manual(values = c("CONTROL" = viridis(3)[2], "PATIENT-DEL" =
     ↪  viridis(3)[1], "PATIENT-DUP" = "orange"), labels = c("Control", "22qDel",
     ↪  "22qDup")) +
    #scale_x_continuous(limits = xlim, expand = c(0,0))+
    #scale_y_continuous(limits = ylim, expand = c(0,0))+
    theme_classic() +
    theme(legend.title = element_blank())+
    theme(axis.title.y = element_text(angle = 0, vjust= 0.5))+
    ylab(ylab)+
    xlab(xlab)+
    #ggtitle(paste(gsub("_", " ", gsub(".combat", "", name)),"Volume"))
    ggtitle(plot_title)
}

#plot_gamm_gdiff(list=mat_full_gdiff, name = "Hip_CA3.combat.norm", xlab= "",
 ↪  ylab= "", title = "CA2/3 Hippocampus")
```

plot age gamms with partial resids for full sample

```
#plot chosen gamms
thal_av <- plot_gamm_resid(list=maturation_full, name = "Thal_AV.combat.norm",
 ↪  xlab= "Age", ylab= "Normalized\nVolume", title = "Anteroventral Thalamus")
hip_ca3 <- plot_gamm_resid(list=maturation_full, name = "Hip_CA3.combat.norm",
 ↪  xlab= "", ylab= "", title = "CA2/3 Hippocampus")
amy_whole <- plot_gamm_resid(list=maturation_full, name =
 ↪  "Amy_Whole_amygdala.combat.norm", xlab= "", ylab= "", title = "Whole
 ↪  Amygdala")
thal_mgn <- plot_gamm_resid(list=maturation_full, name = "Thal_MGN.combat.norm",
 ↪  xlab= "", ylab= "", title = "Medial Geniculate Thalamus")
```

```
thal_mvre <- plot_gamm_resid(list=maturation_full, name =
 ↪  "Thal_MV_Re_.combat.norm", xlab= "", ylab= "", title = "Medial Ventral
 ↪  Thalamus")
hip_ca4 <- plot_gamm_resid(list=maturation_full, name = "Hip_CA4.combat.norm",
 ↪  xlab= "", ylab= "", title = "CA4 Hippocampus")
#amy_aaa <- plot_gamm_resid(name = "Amy_Anterior_amygdaloid_area_AAA.combat.norm",
 ↪  xlab= "Age", ylab= "Volume")
#thal_lgn <- plot_gamm_resid(name = "Thal_LGN.combat.norm", xlab= "Age", ylab=
 ↪  "Volume")
#thal_mgn <- plot_gamm_resid(name = "Thal_MGN.combat.norm", xlab= "Age", ylab=
 ↪  "Volume")
#hip_ca1 <- plot_gamm_resid(name = "Hip_CA1.combat.norm", xlab= "", ylab= "",
 ↪  title = "CA1 Hippocampus")


plot_age <- (thal_av + hip_ca3 + hip_ca4) / (thal_mgn + thal_mvre + amy_whole) +
 ↪  plot_layout(guides = "collect") + plot_annotation(tag_levels = 'A', title =
 ↪  "Developmental Trajectories\nFull age range") &   theme(plot.tag =
 ↪  element_text(face = 'bold')) + theme(legend.position = "bottom",
 ↪  legend.text=element_text(size = 14), plot.title = element_text(size = 16,
 ↪  hjust = 0))

plot_age
```



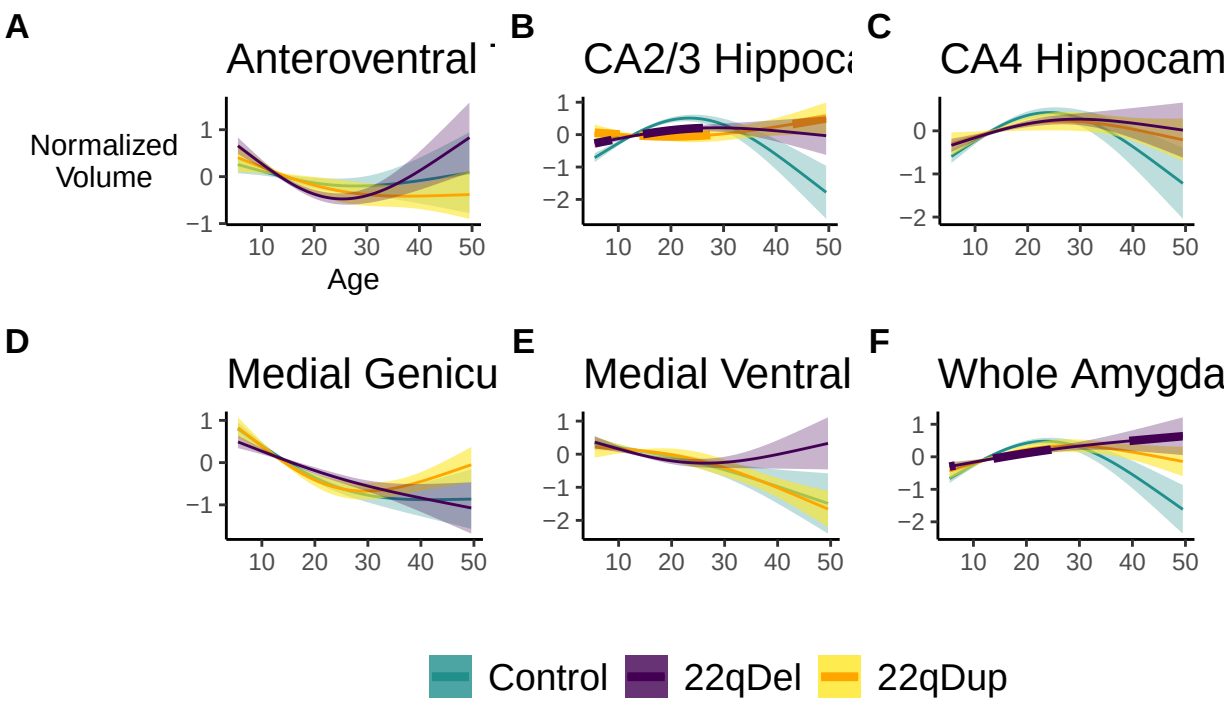

```
#ggsave(plot=plot_age, filename =file.path(project,
 ↪  "figures/age/age_gamms_curves_resid_full.pdf"), width= 12, height= 7, device =
 ↪  "pdf")
```

```
#ggsave(plot=plot_age, filename =file.path(project,
 ↪  "figures/age/age_gamms_curves_resid_full.png"), width= 12, height= 7, device =
 ↪  "png", dpi = 300)
```

plot age gamms with group differences for full sample

```
#plot chosen gamms
thal_av_gd <- plot_gamm_gdiff(list=mat_full_gdiff, name = "Thal_AV.combat.norm",
 ↪  xlab= "Age", ylab= "Normalized\nVolume", title = "Anteroventral Thalamus")
hip_ca3_gd <- plot_gamm_gdiff(list=mat_full_gdiff, name = "Hip_CA3.combat.norm",
 ↪  xlab= "", ylab= "", title = "CA2/3 Hippocampus")
amy_whole_gd <- plot_gamm_gdiff(list=mat_full_gdiff, name =
 ↪  "Amy_Whole_amygdala.combat.norm", xlab= "", ylab= "", title = "Whole
 ↪  Amygdala")
#thal_md_gd <- plot_gamm_gdiff(list=mat_full_gdiff, name =
 ↪  "Thal_MD_all.combat.norm", xlab= "", ylab= "", title = "Mediodorsal Thalamus")
thal_mgn_gd <- plot_gamm_gdiff(list=mat_full_gdiff, name = "Thal_MGN.combat.norm",
 ↪  xlab= "", ylab= "", title = "Medial Geniculate Thalamus")
thal_mvre_gd <- plot_gamm_gdiff(list=mat_full_gdiff, name =
 ↪  "Thal_MV_Re_.combat.norm", xlab= "", ylab= "", title = "Medial Ventral
 ↪  Thalamus")
hip_ca4_gd <- plot_gamm_gdiff(list=mat_full_gdiff, name = "Hip_CA4.combat.norm",
 ↪  xlab= "", ylab= "", title = "CA4 Hippocampus")

plot_age_gd <- (thal_av_gd + hip_ca3_gd + hip_ca4_gd) / (thal_mgn_gd +
 ↪  thal_mvre_gd + amy_whole_gd) + plot_layout(guides = "collect") +
 ↪  plot_annotation(tag_levels = 'A', title = "Developmental Trajectories\nFull
 ↪  age range") &   theme(plot.tag = element_text(face = 'bold')) +
 ↪  theme(legend.position = "bottom", legend.text=element_text(size = 14),
 ↪  plot.title = element_text(size = 16, hjust = 0))

plot_age_gd
```

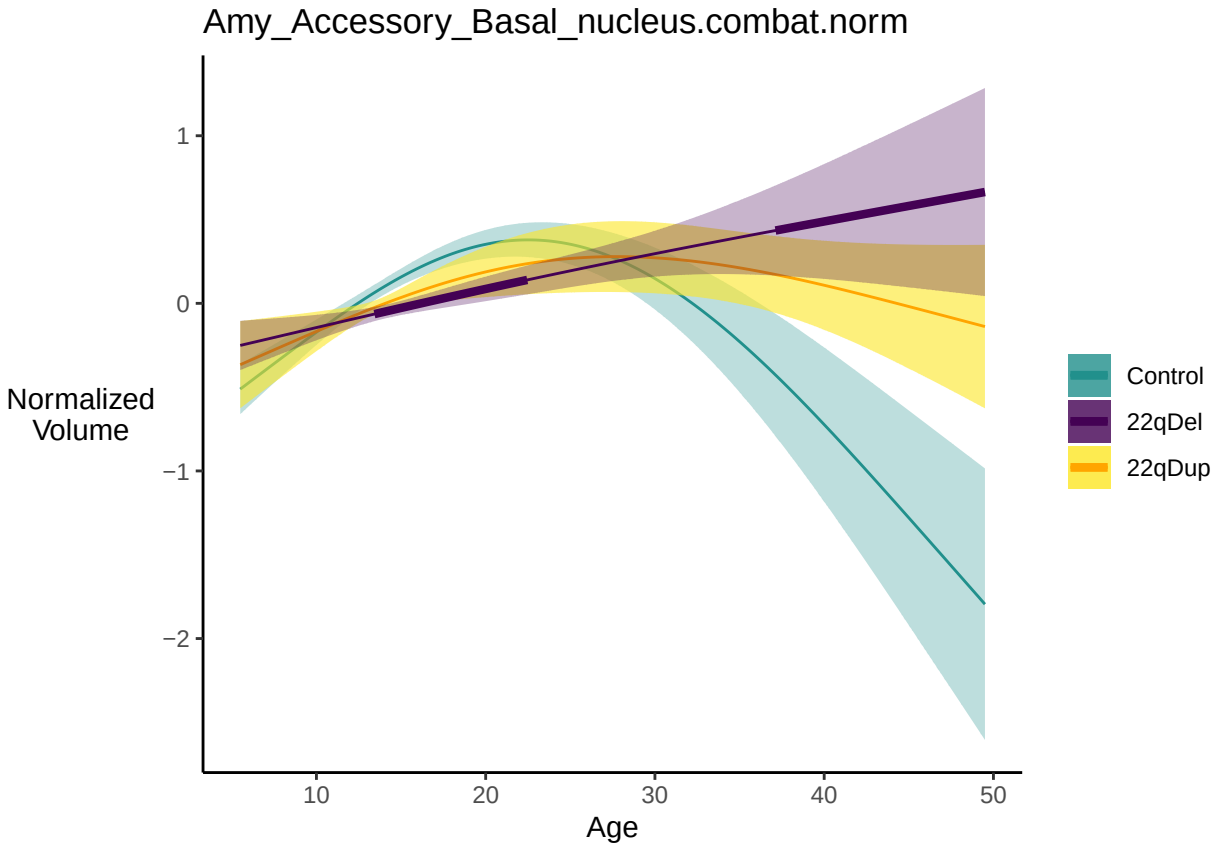
Developmental Trajectories
Full age range

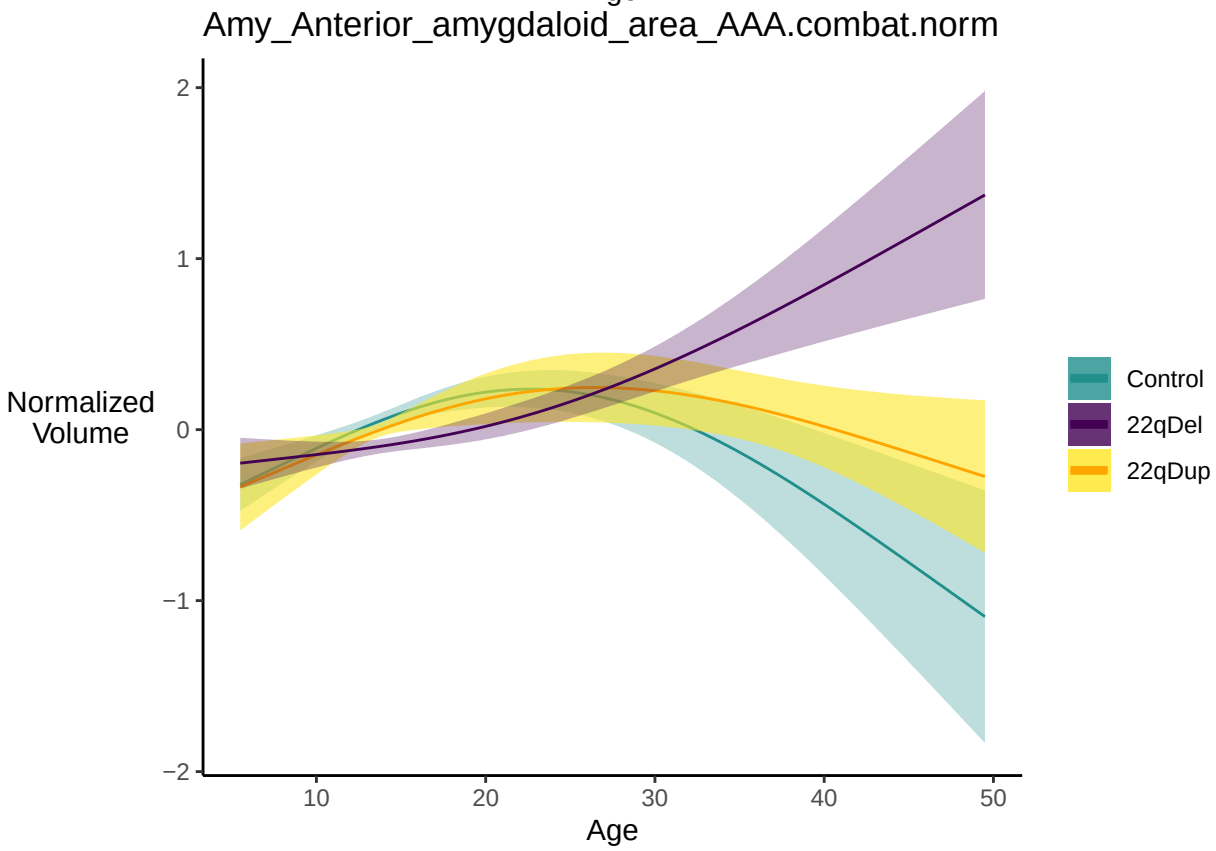


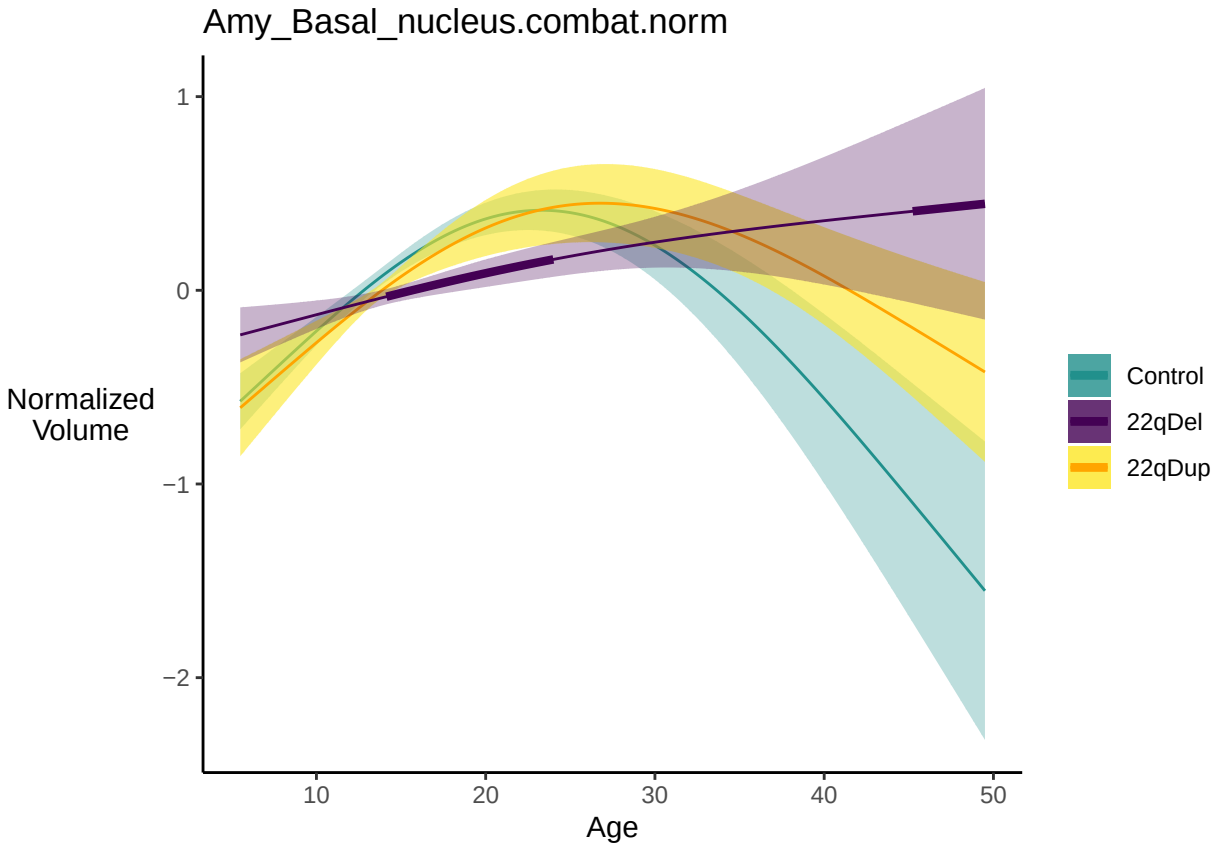

```
#ggsave(plot=plot_age_gd, filename =file.path(project,
 ↪  "figures/age/age_gamms_curves_gdiff_full.pdf"), width= 12, height= 7, device =
 ↪  "pdf")
#ggsave(plot=plot_age_gd, filename =file.path(project,
 ↪  "figures/age/age_gamms_curves_gdiff_full.png"), width= 12, height= 7, device =
 ↪  "png", dpi = 300)
```

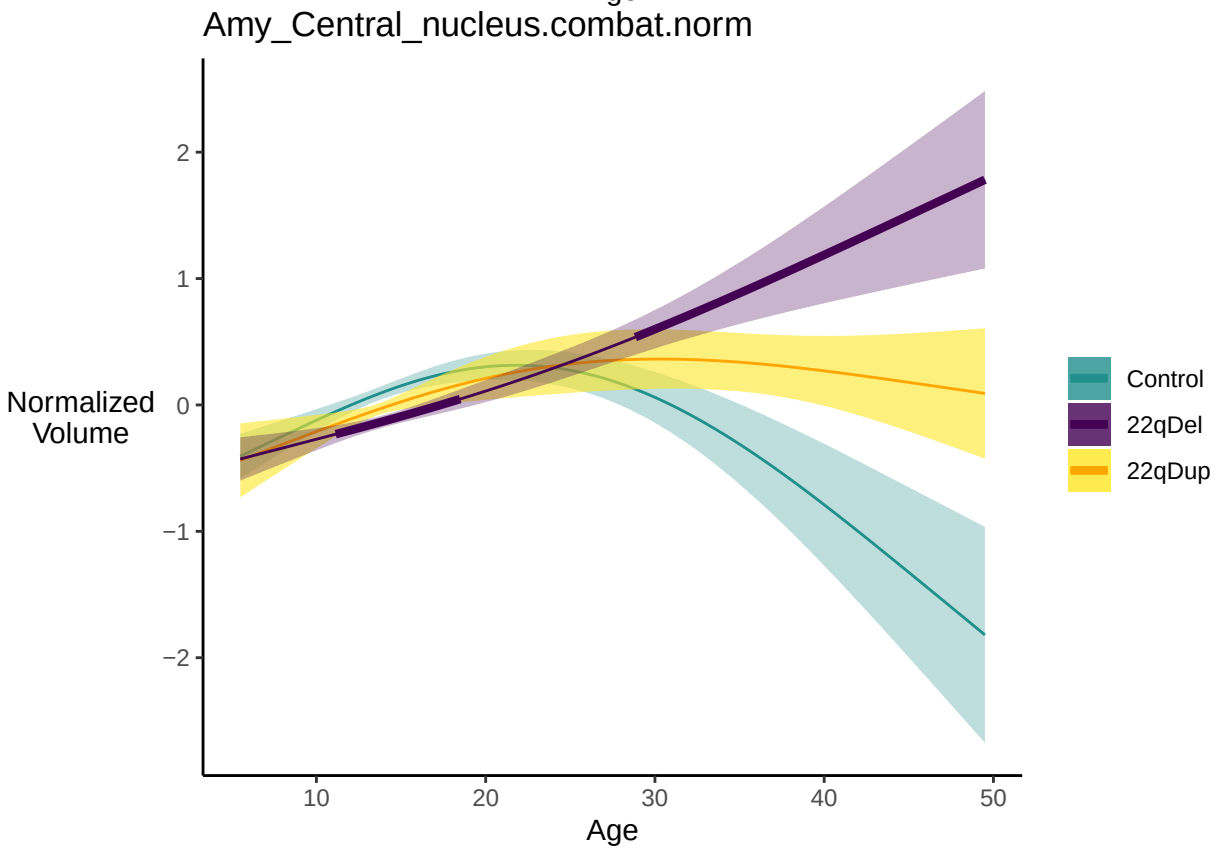
check some additional ones

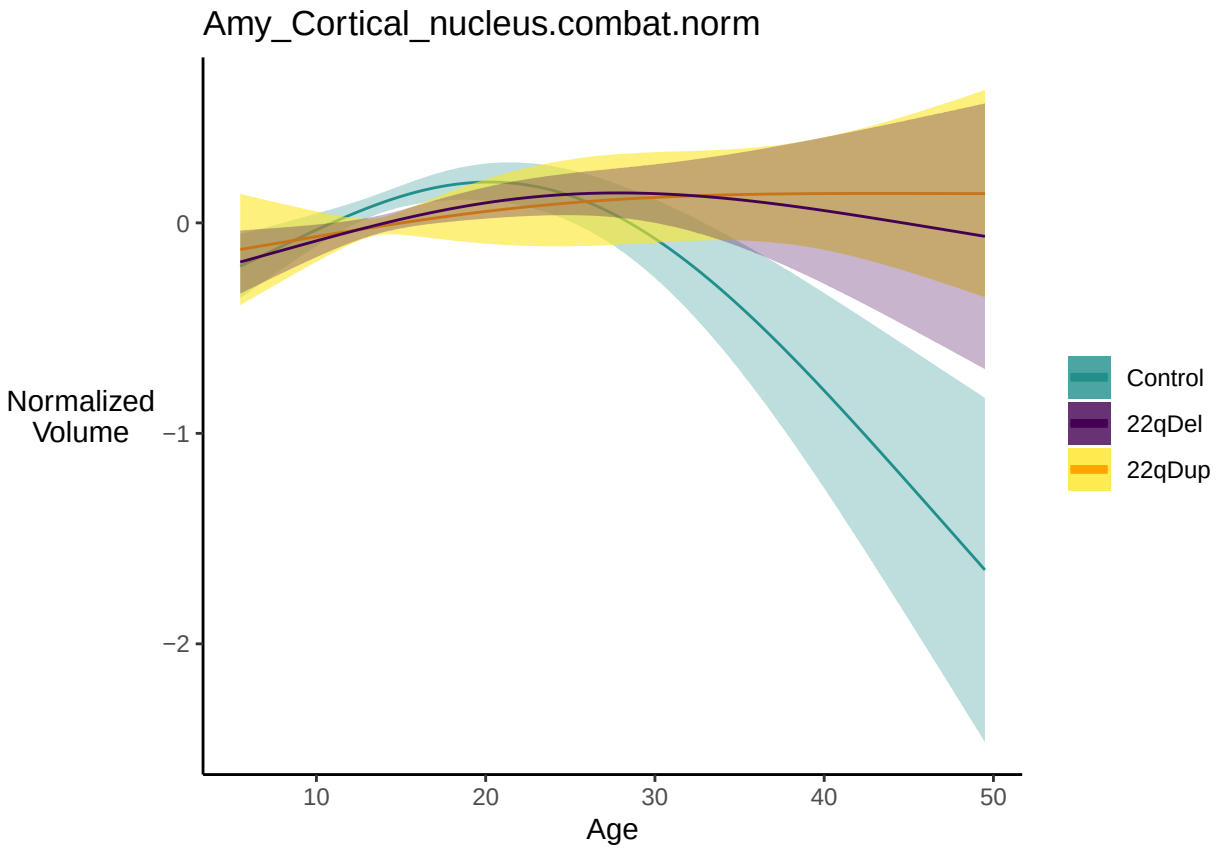
```
lapply(all_names_normed, function(n) plot_gamm_gdiff(list=mat_full_gdiff, name =n,
 ↪  xlab= "Age", ylab= "Normalized\nVolume", title =n))
```

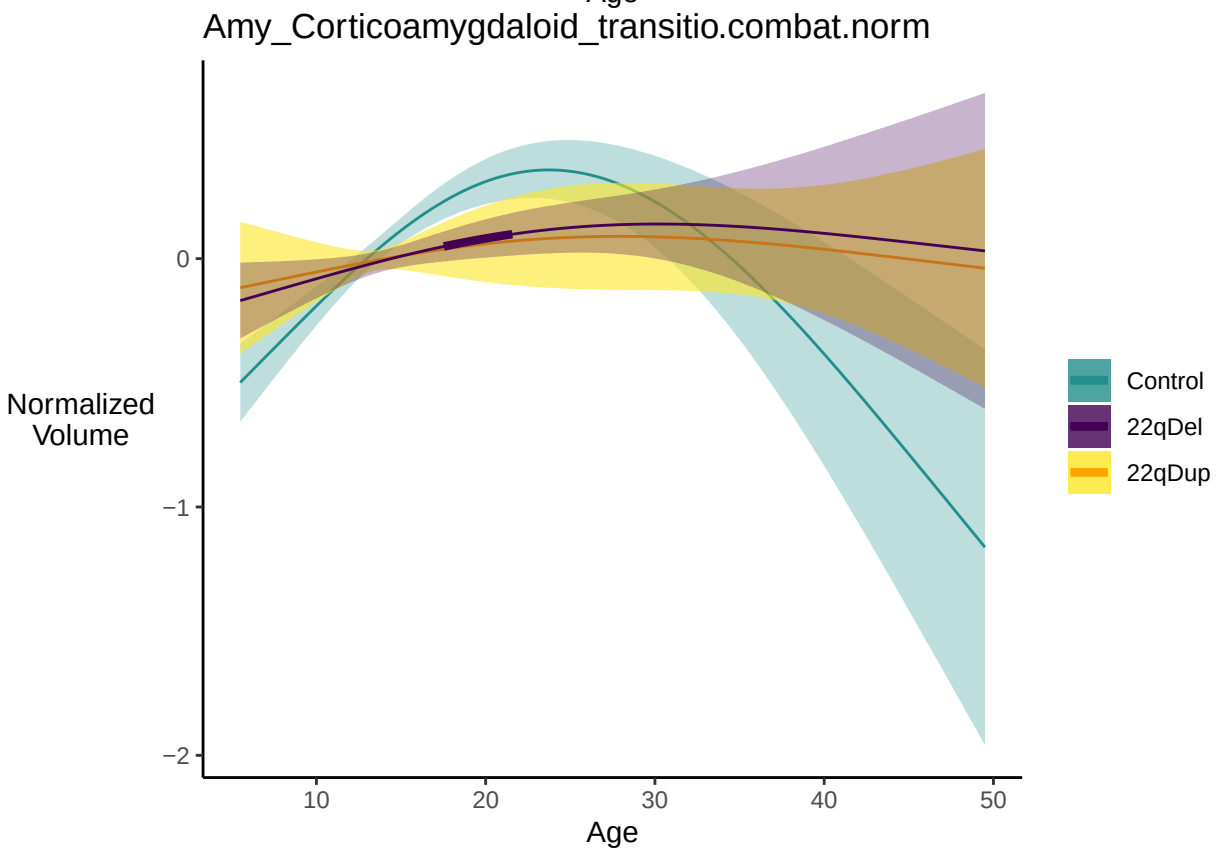

Amy_Accessory_Basal_nucleus.combat.norm
Normalized Volume
1
0
−1
−2
10
20
30
40
50
Age
Control
22qDel
22qDup

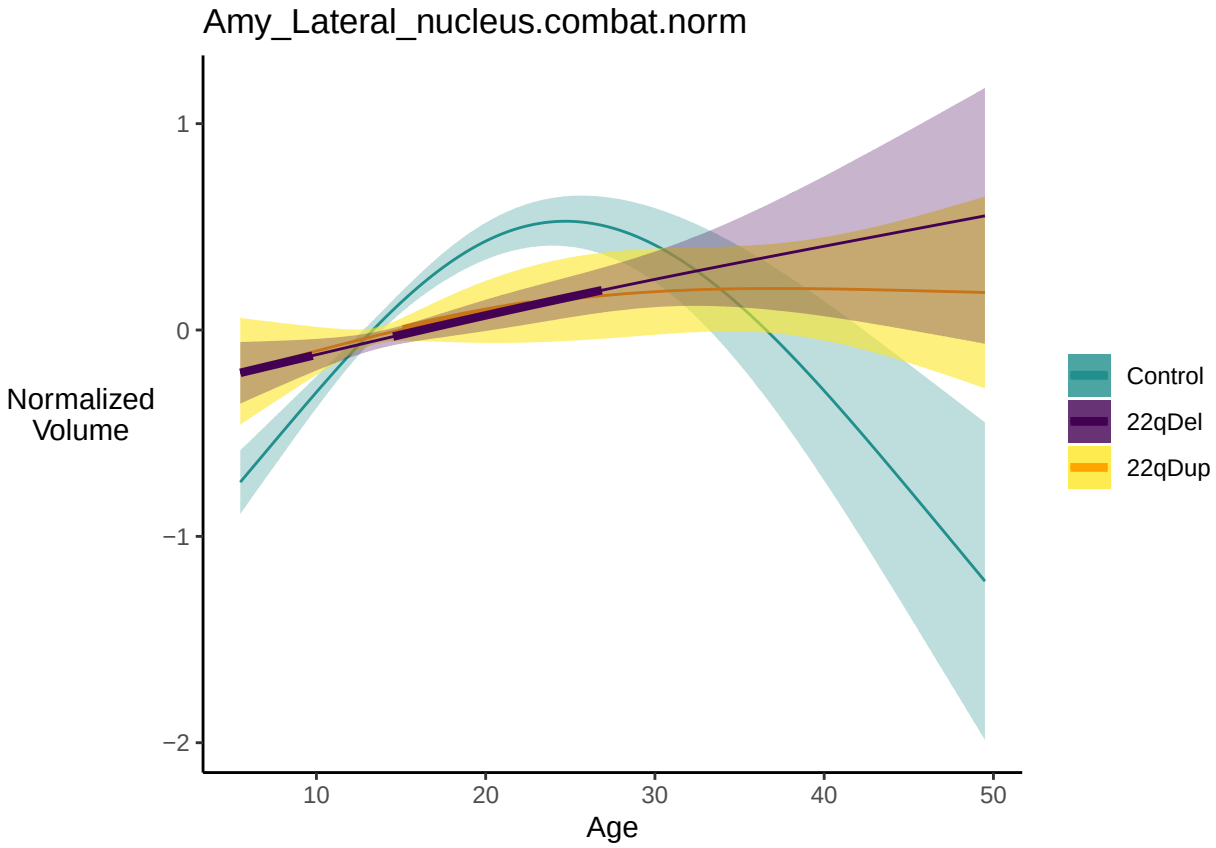

Amy_Anterior_amygdaloid_area_AAA.combat.norm
Normalized Volume
2
1
0
−1
−2
10
20
30
40
50
Age
Control
22qDel
22qDup

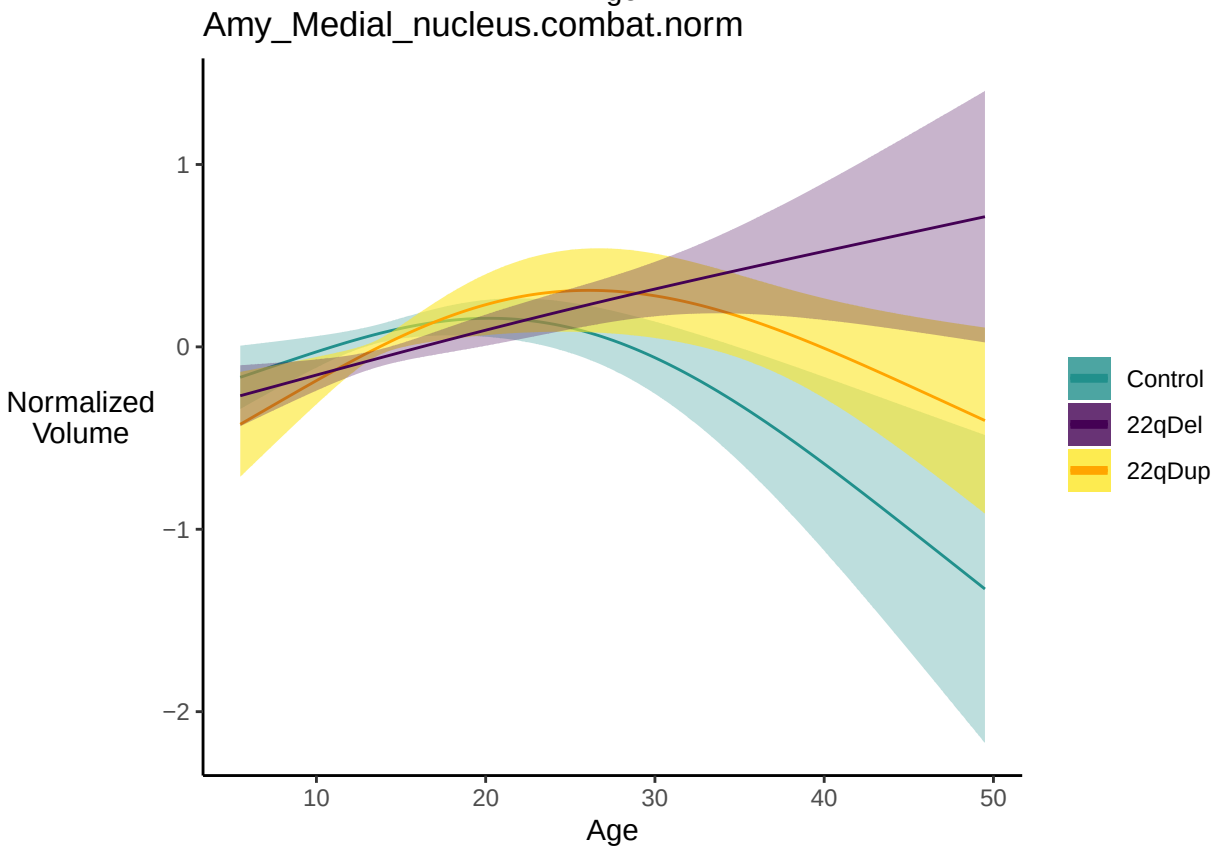

Amy_Basal_nucleus.combat.norm
Normalized Volume
1
0
−1
−2
10
20
30
40
50
Age
Control
22qDel
22qDup

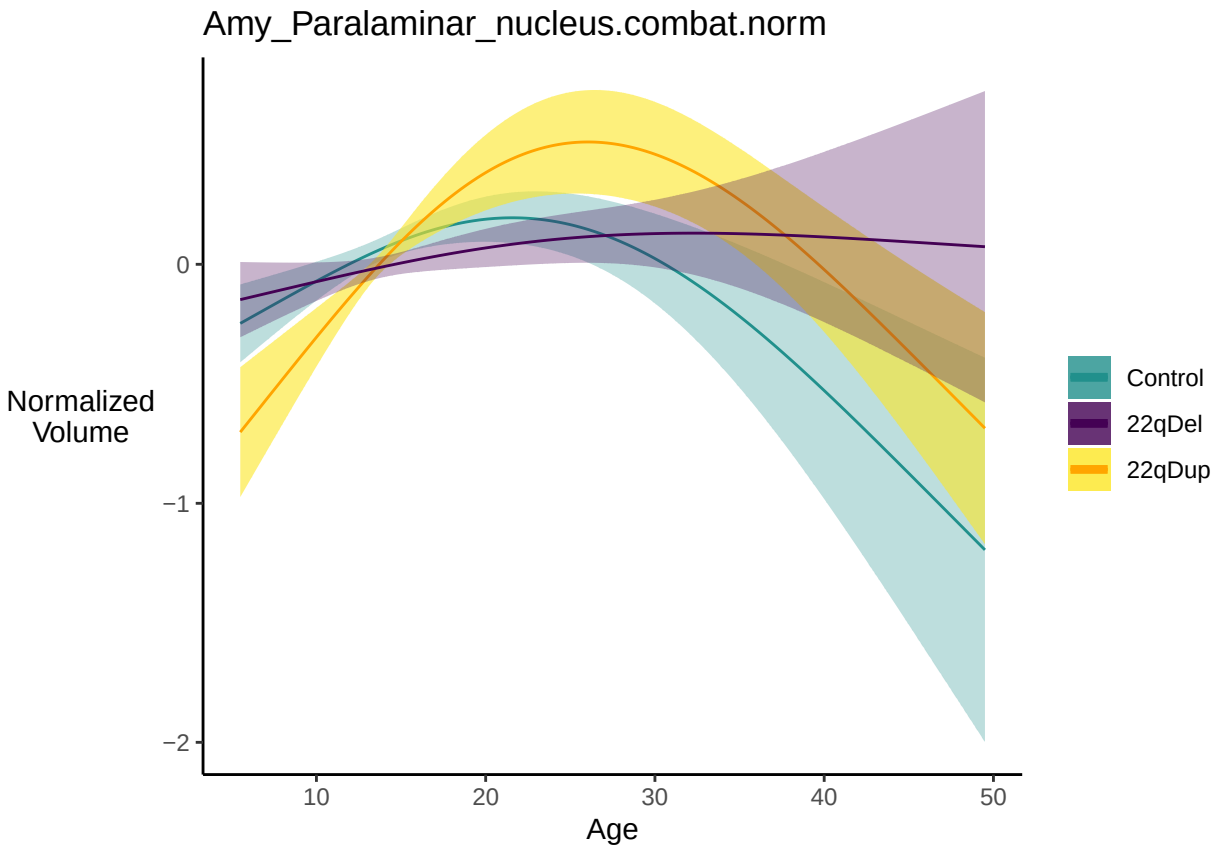

Amy_Central_nucleus.combat.norm
Normalized Volume
2
1
0
−1
−2
10
20
30
40
50
Age
Control
22qDel
22qDup

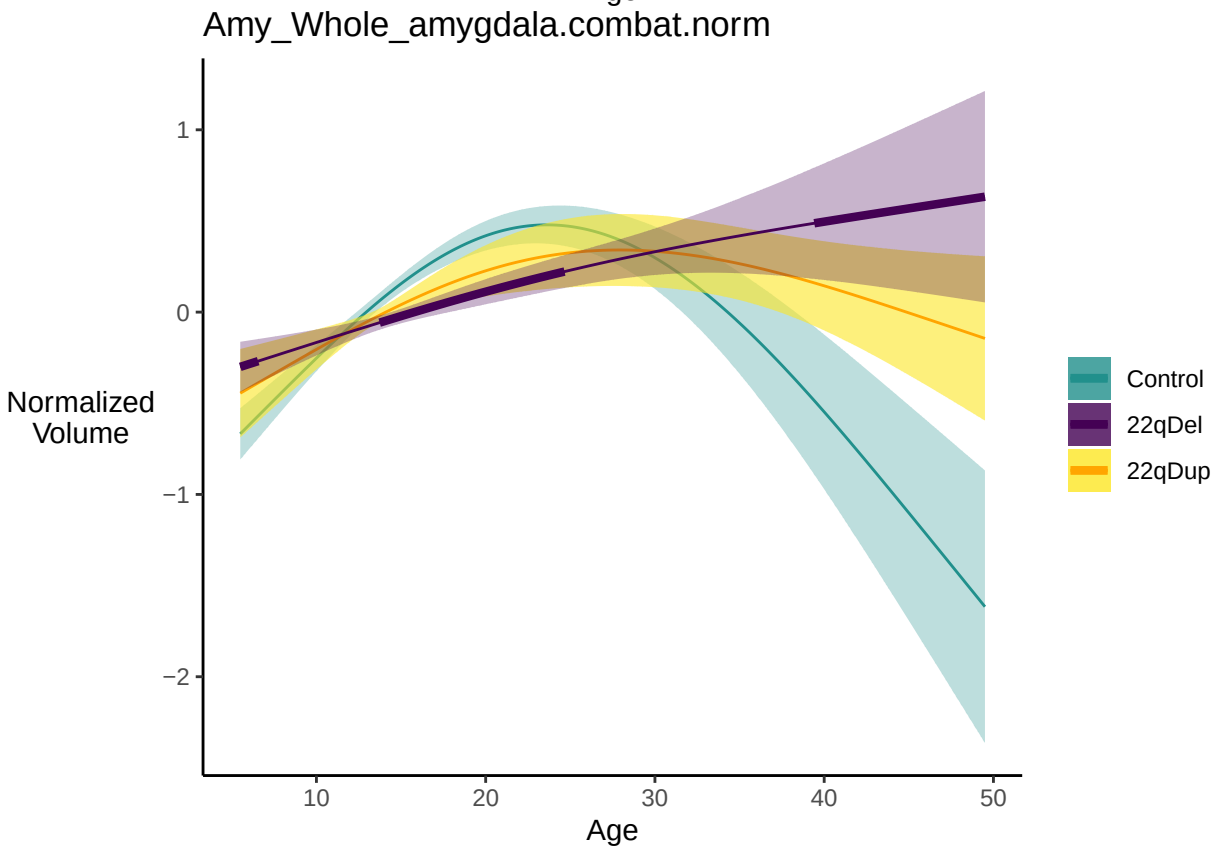

Amy_Cortical_nucleus.combat.norm
Normalized Volume
0
−1
−2
10
20
30
40
50
Age
Control
22qDel
22qDup

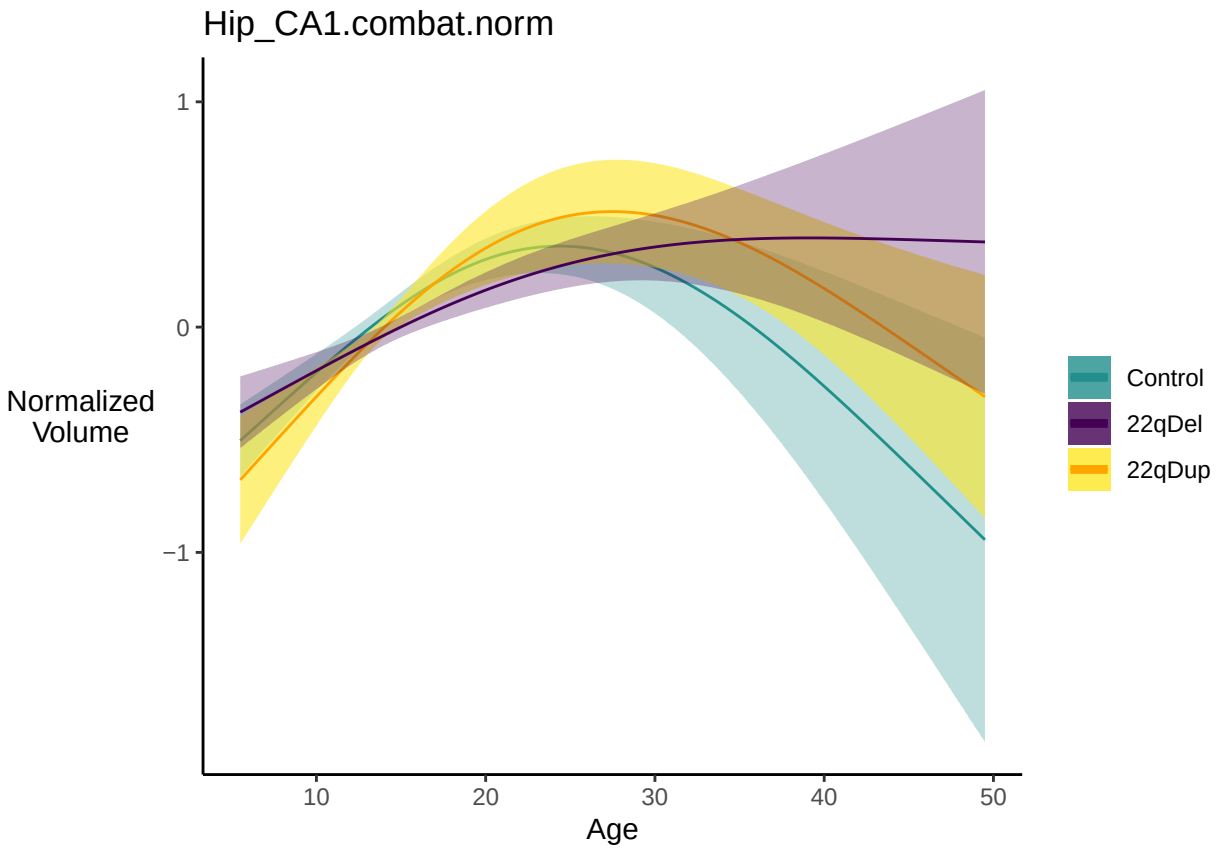

Amy_Corticoamygdaloid_transitio.combat.norm
Normalized Volume
0
−1
−2
10
20
30
40
50
Age
Control
22qDel
22qDup

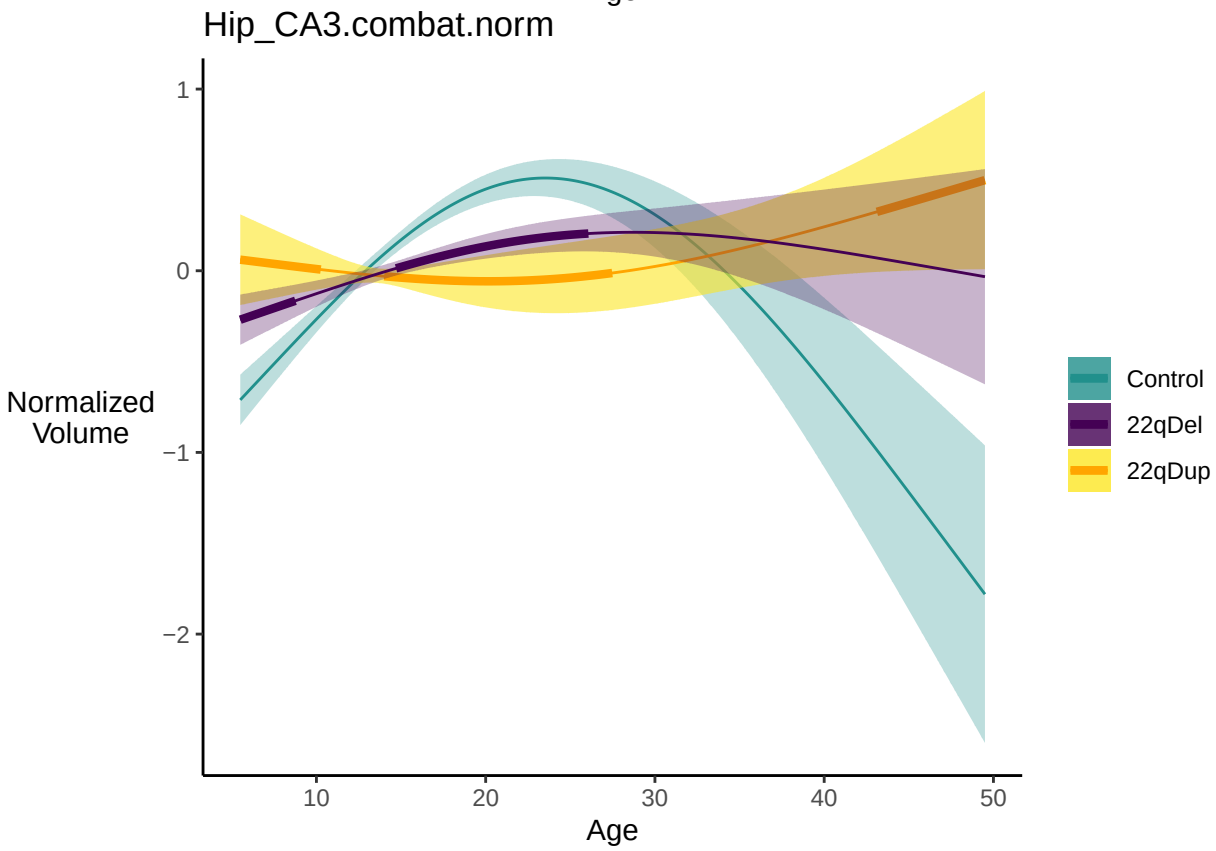

Amy_Lateral_nucleus.combat.norm
Normalized Volume
1
0
−1
−2
10
20
30
40
50
Age
Control
22qDel
22qDup

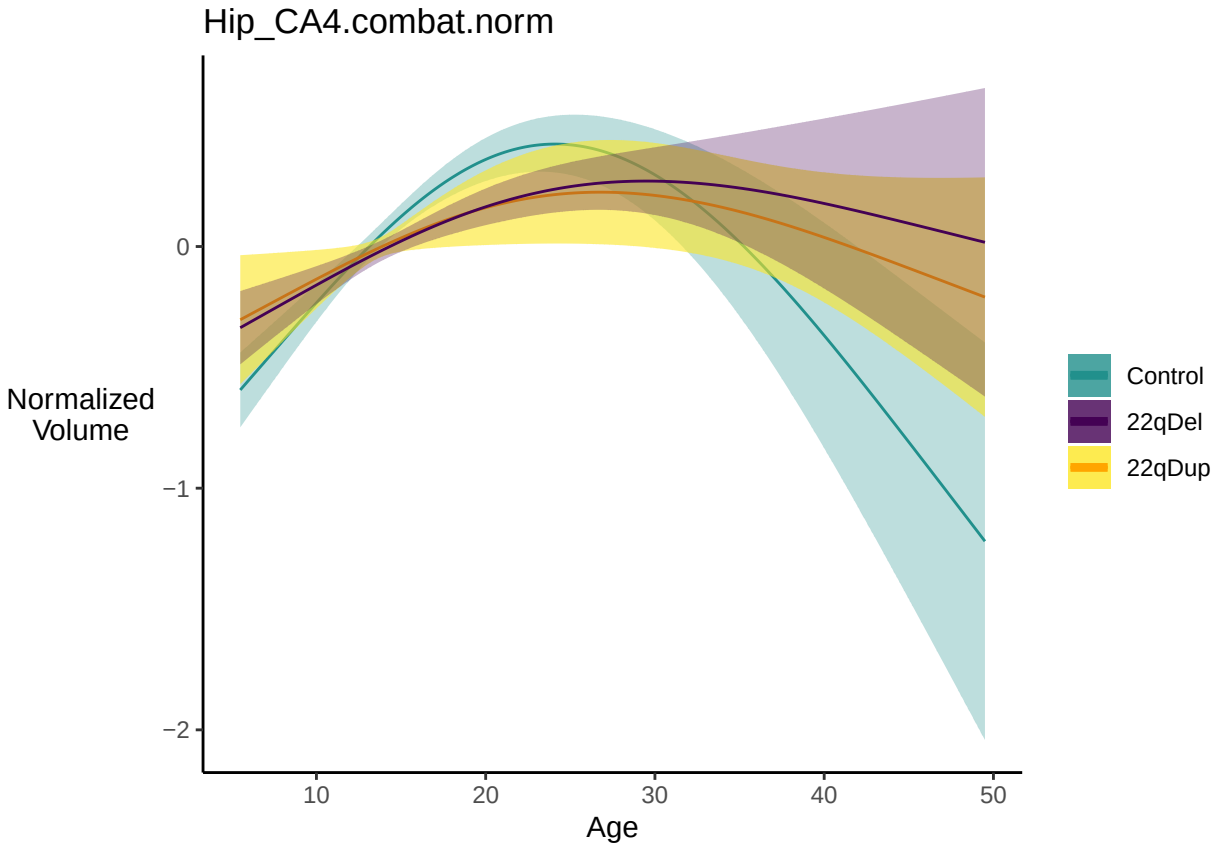

Amy_Medial_nucleus.combat.norm
Normalized Volume
1
0
−1
−2
10
20
30
40
50
Age
Control
22qDel
22qDup

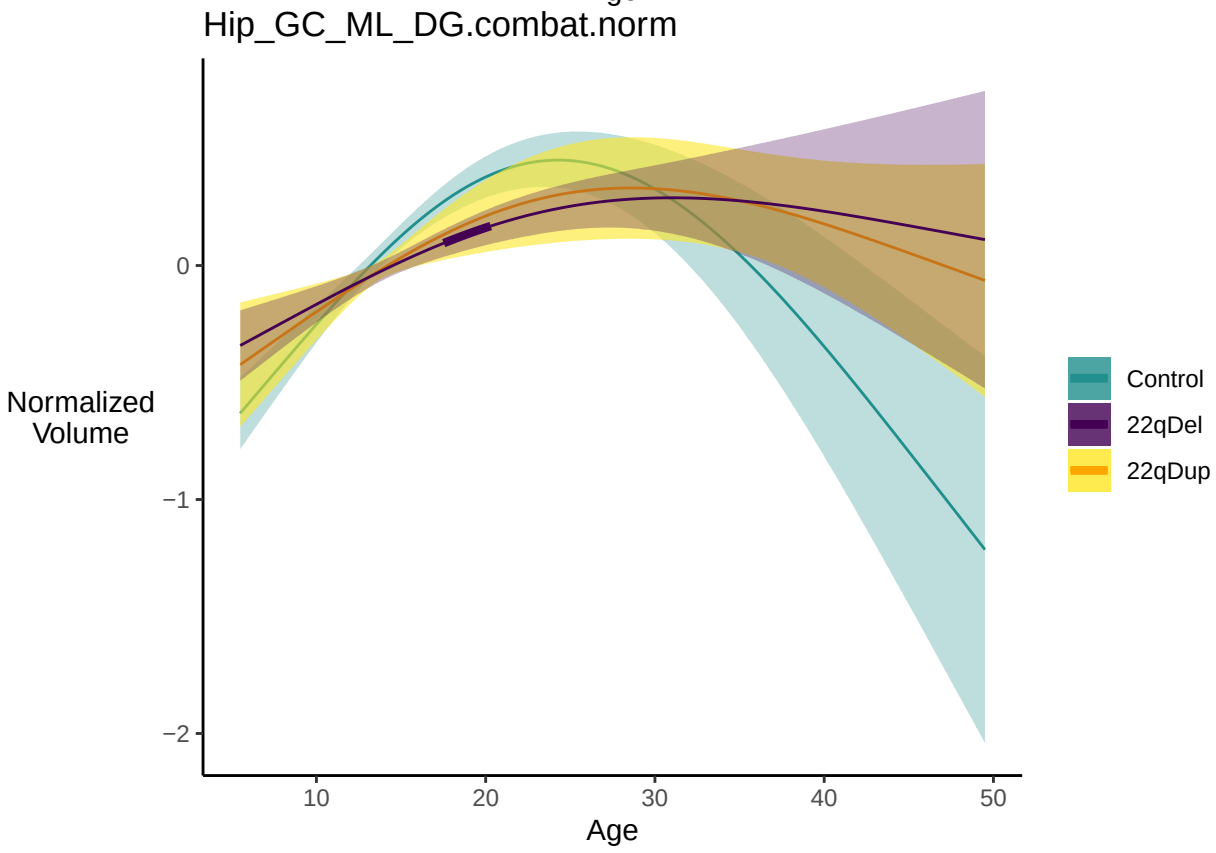

Amy_Paralaminar_nucleus.combat.norm
Normalized Volume
0
−1
−2
10
20
30
40
50
Age
Control
22qDel
22qDup

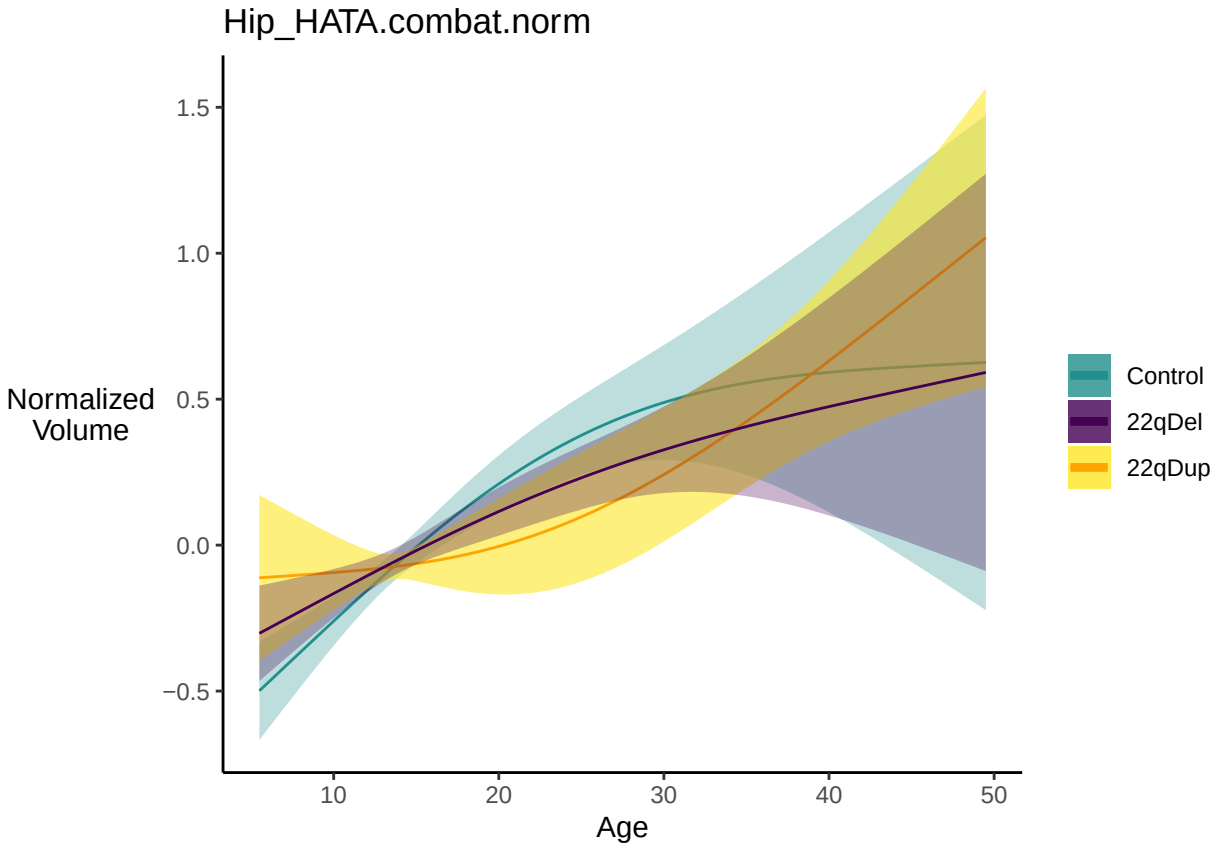

Amy_Whole_amygdala.combat.norm
Normalized Volume
1
0
−1
−2
10
20
30
40
50
Age
Control
22qDel
22qDup

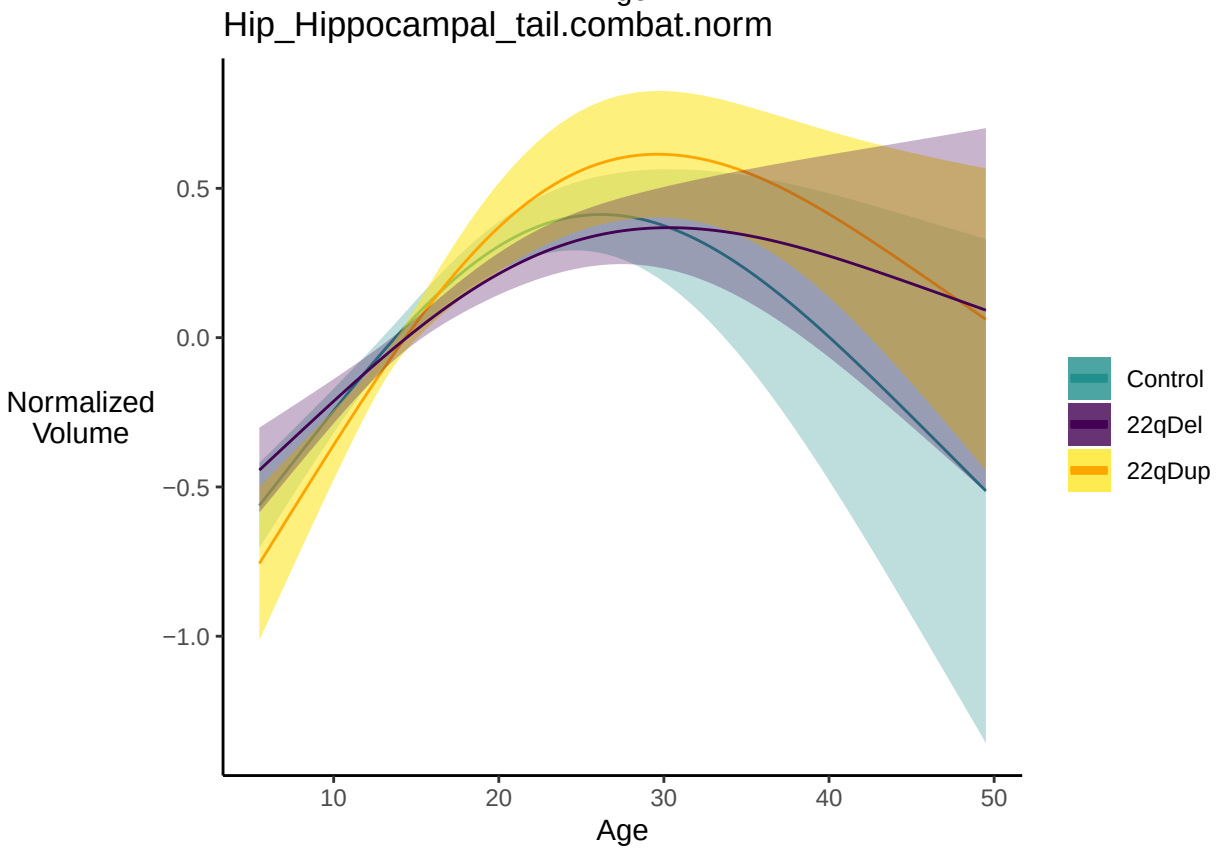

Hip_CA1.combat.norm
Normalized Volume
1
0
−1
10
20
30
40
50
Age
Control
22qDel
22qDup

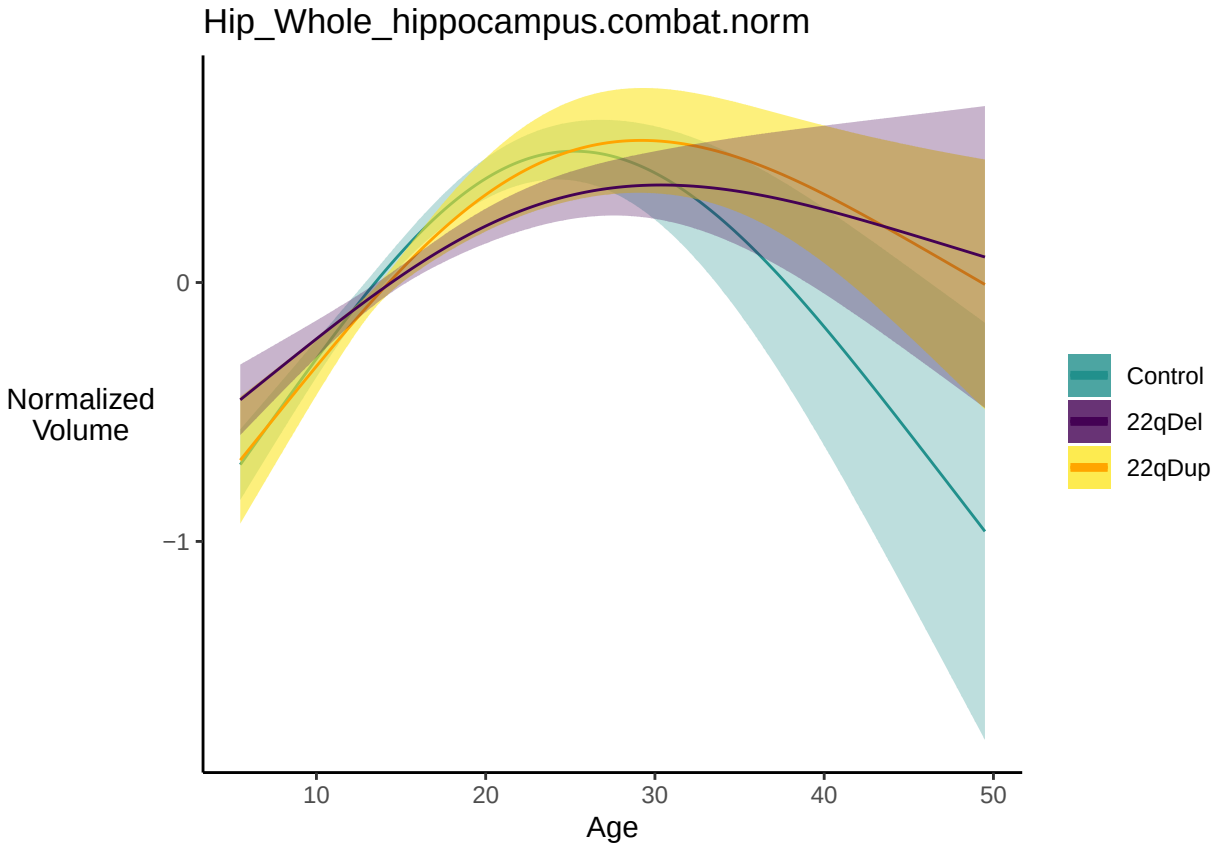

Hip_CA3.combat.norm
Normalized Volume
1
0
−1
−2
10
20
30
40
50
Age
Control
22qDel
22qDup

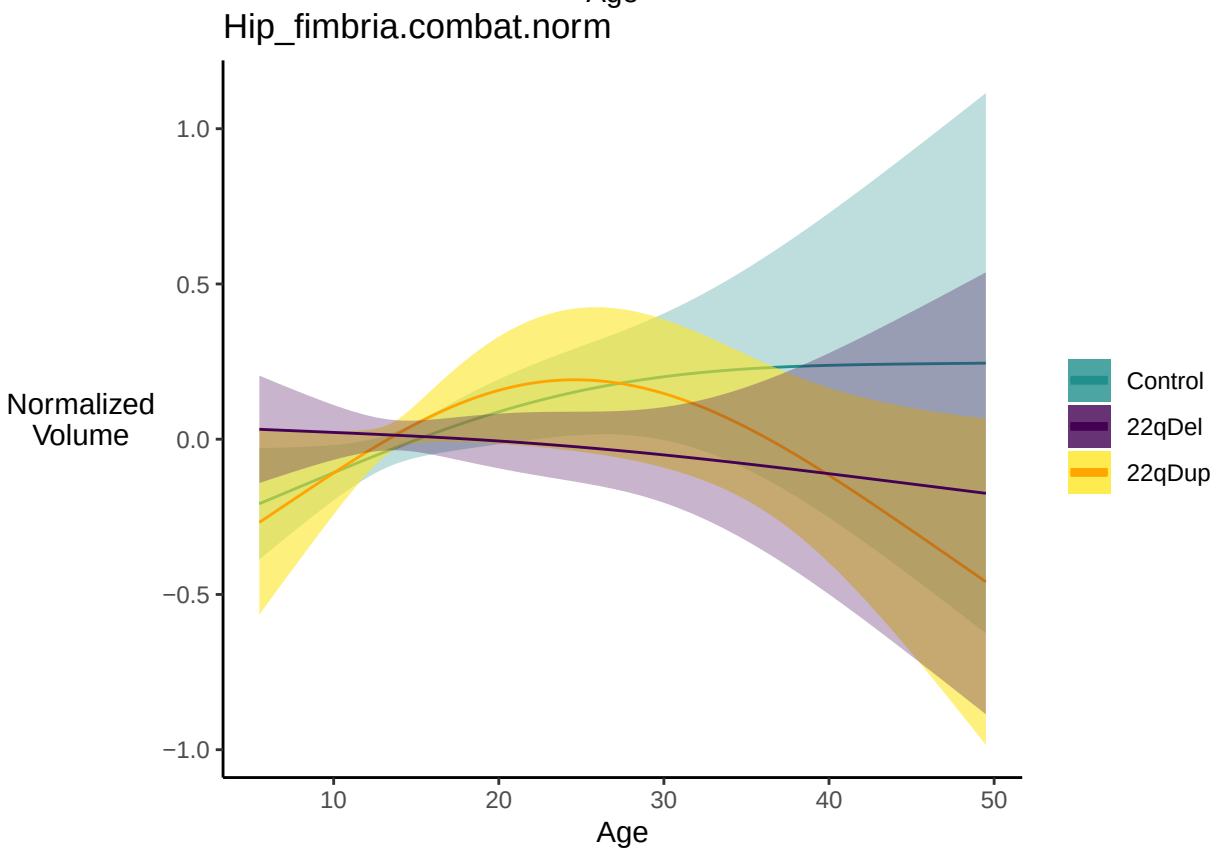

Hip_CA4.combat.norm
Normalized Volume
0
−1
−2
10
20
30
40
50
Age
Control
22qDel
22qDup

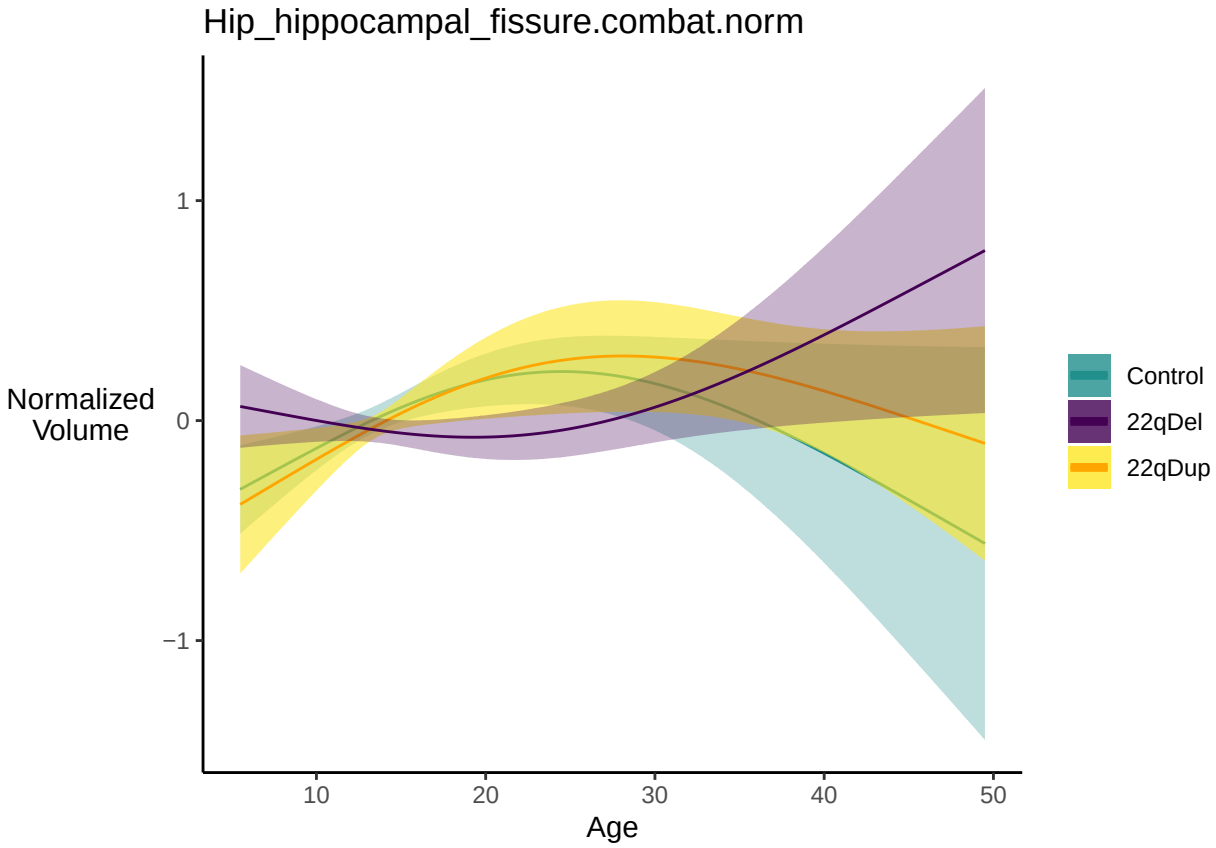

Hip_GC_ML_DG.combat.norm
Normalized Volume
0
−1
−2
10
20
30
40
50
Age
Control
22qDel
22qDup

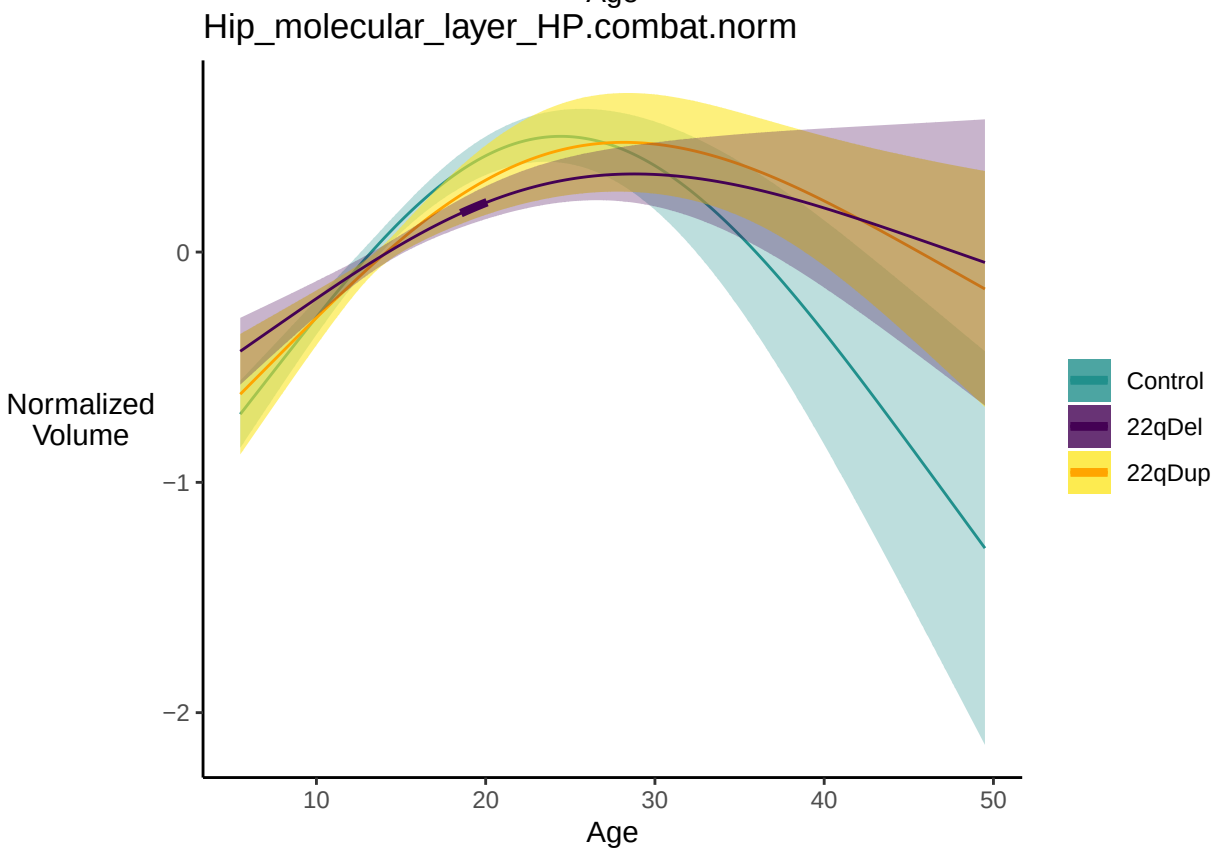

Hip_HATA.combat.norm
Normalized Volume
1.5
1.0
0.5
0.0
−0.5
10
20
30
40
50
Age
Control
22qDel
22qDup

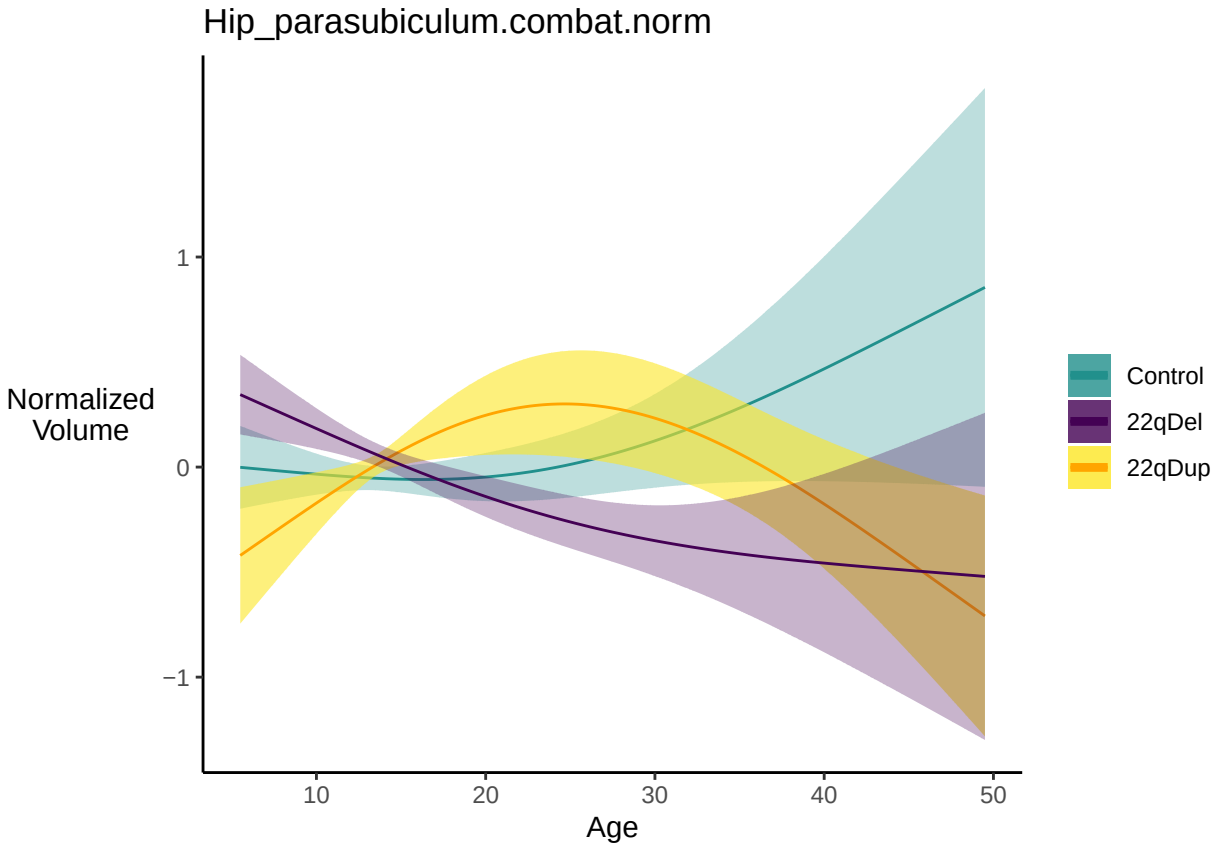

Hip_Hippocampal_tail.combat.norm
Normalized Volume
0.5
0.0
−0.5
−1.0
10
20
30
40
50
Age
Control
22qDel
22qDup

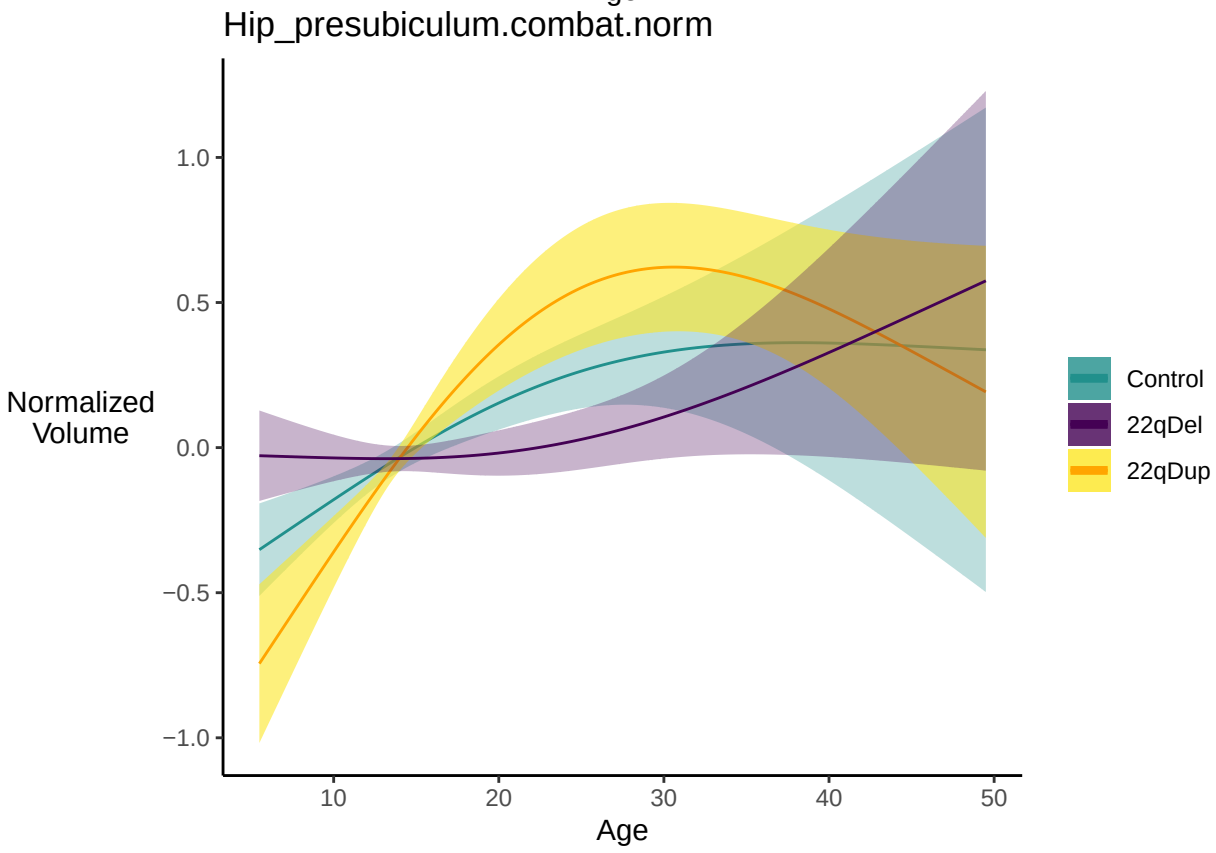

Hip_Whole_hippocampus.combat.norm
Normalized Volume
0
−1
10
20
30
40
50
Age
Control
22qDel
22qDup

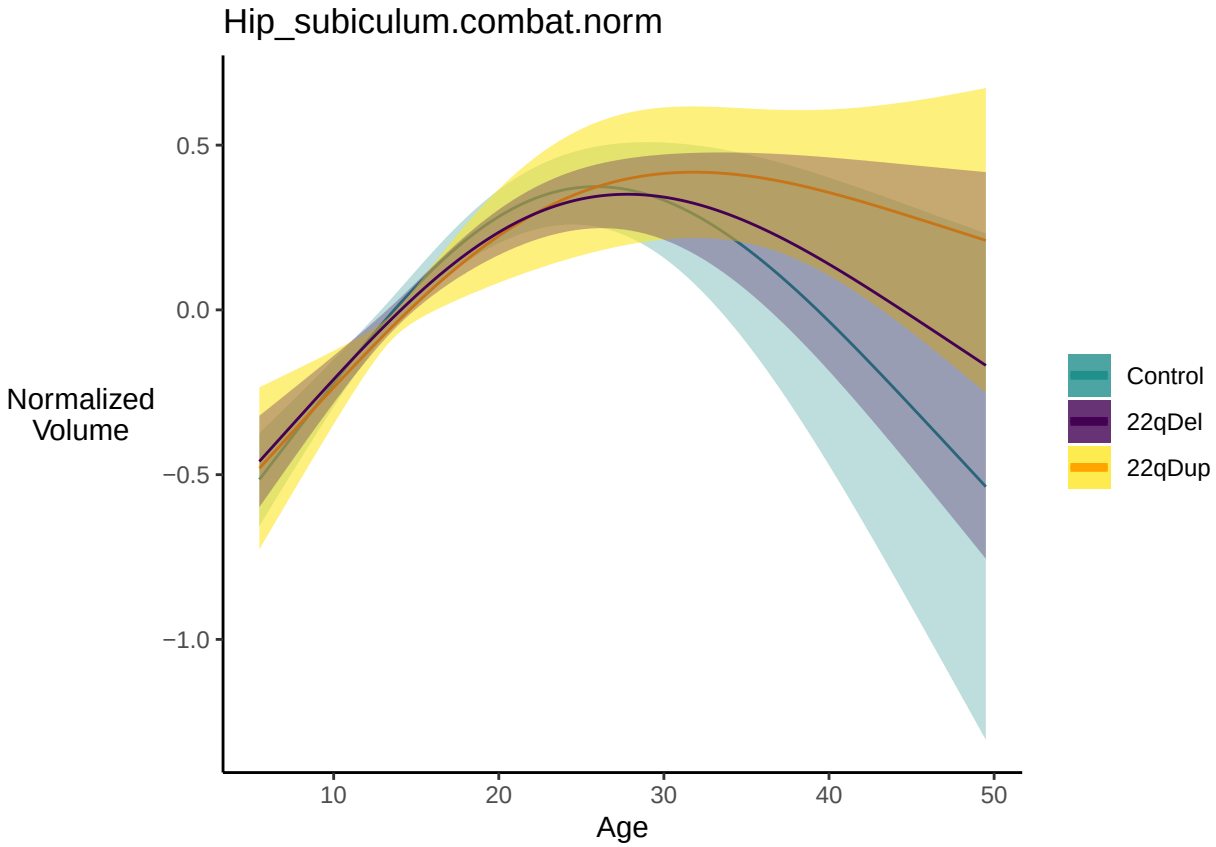

Hip_fimbria.combat.norm
Normalized Volume
1.0
0.5
0.0
−0.5
−1.0
10
20
30
40
50
Age
Control
22qDel
22qDup

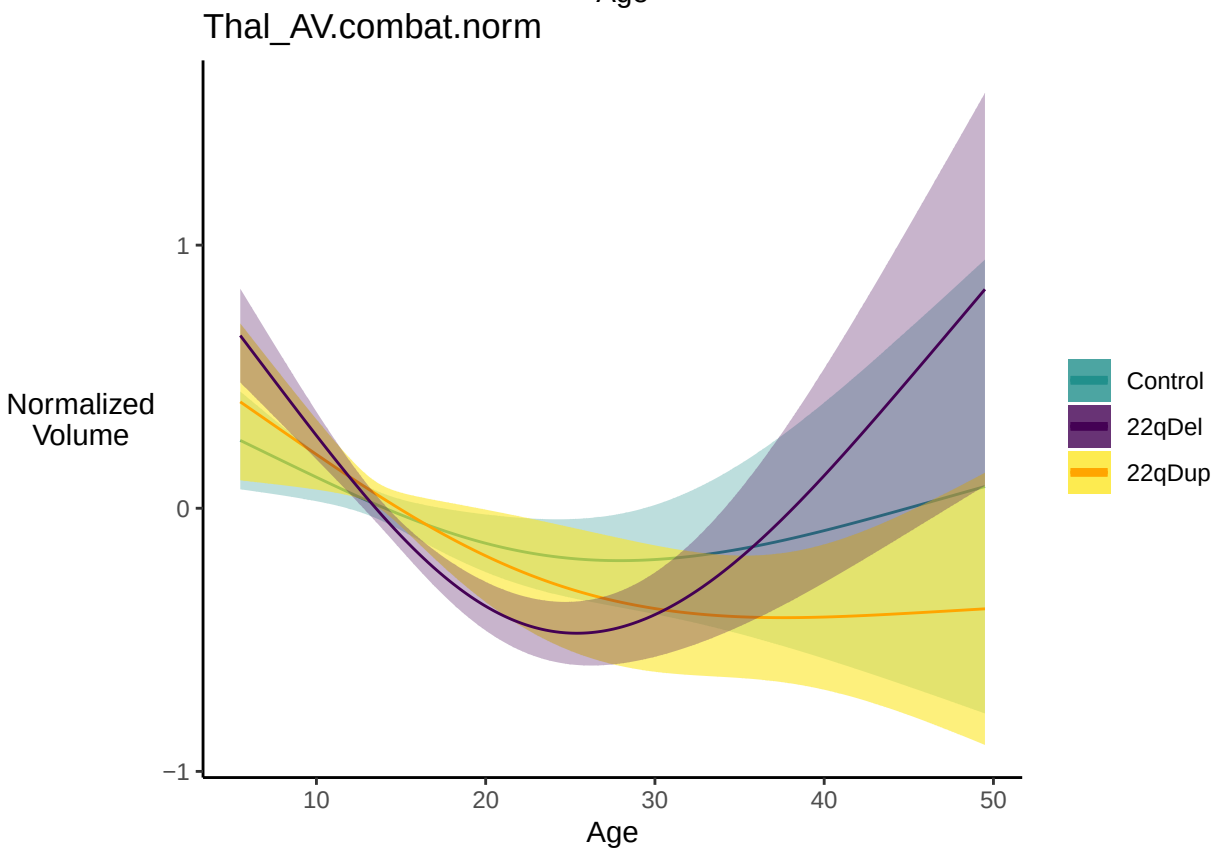

Hip_hippocampal_fissure.combat.norm
Normalized Volume
1
0
−1
10
20
30
40
50
Age
Control
22qDel
22qDup

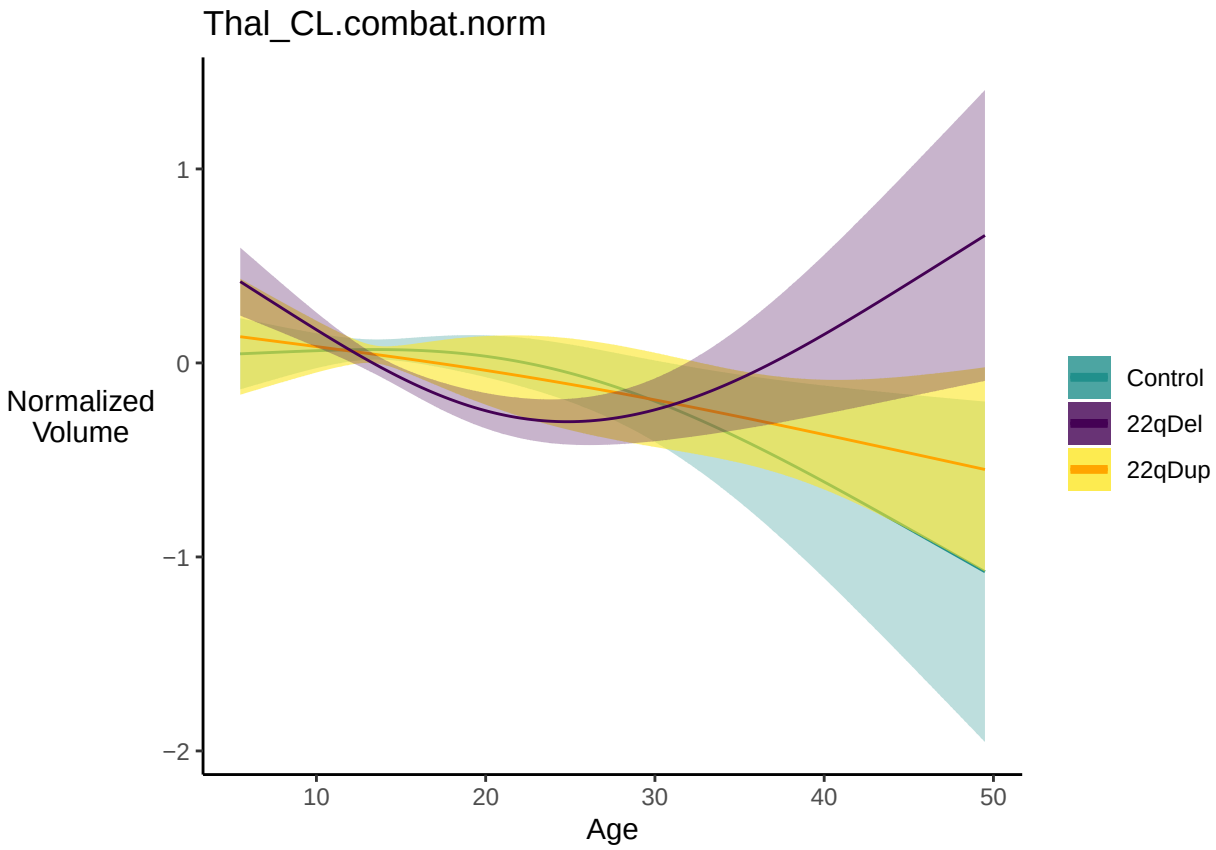

Hip_molecular_layer_HP.combat.norm
Normalized Volume
0
−1
−2
10
20
30
40
50
Age
Control
22qDel
22qDup

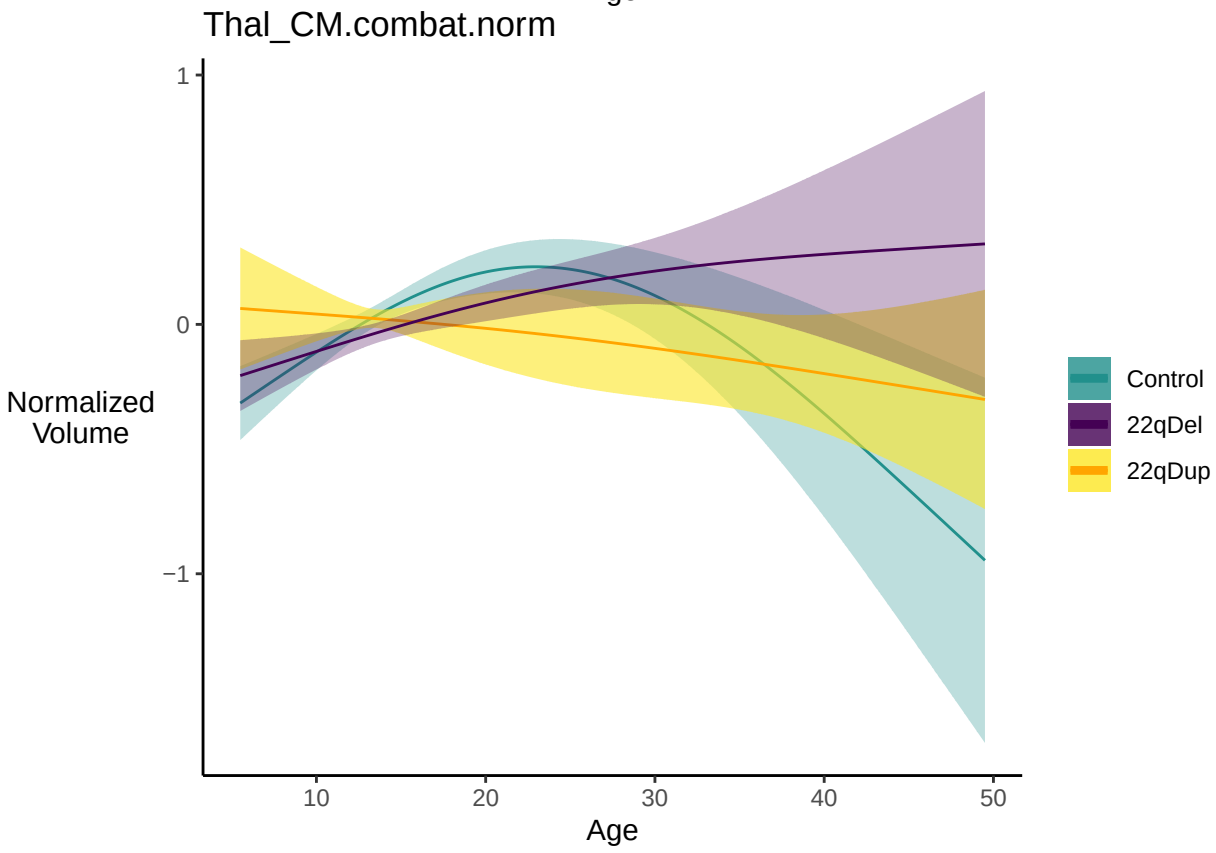

Hip_parasubiculum.combat.norm
Normalized Volume
1
0
−1
10
20
30
40
50
Age
Control
22qDel
22qDup

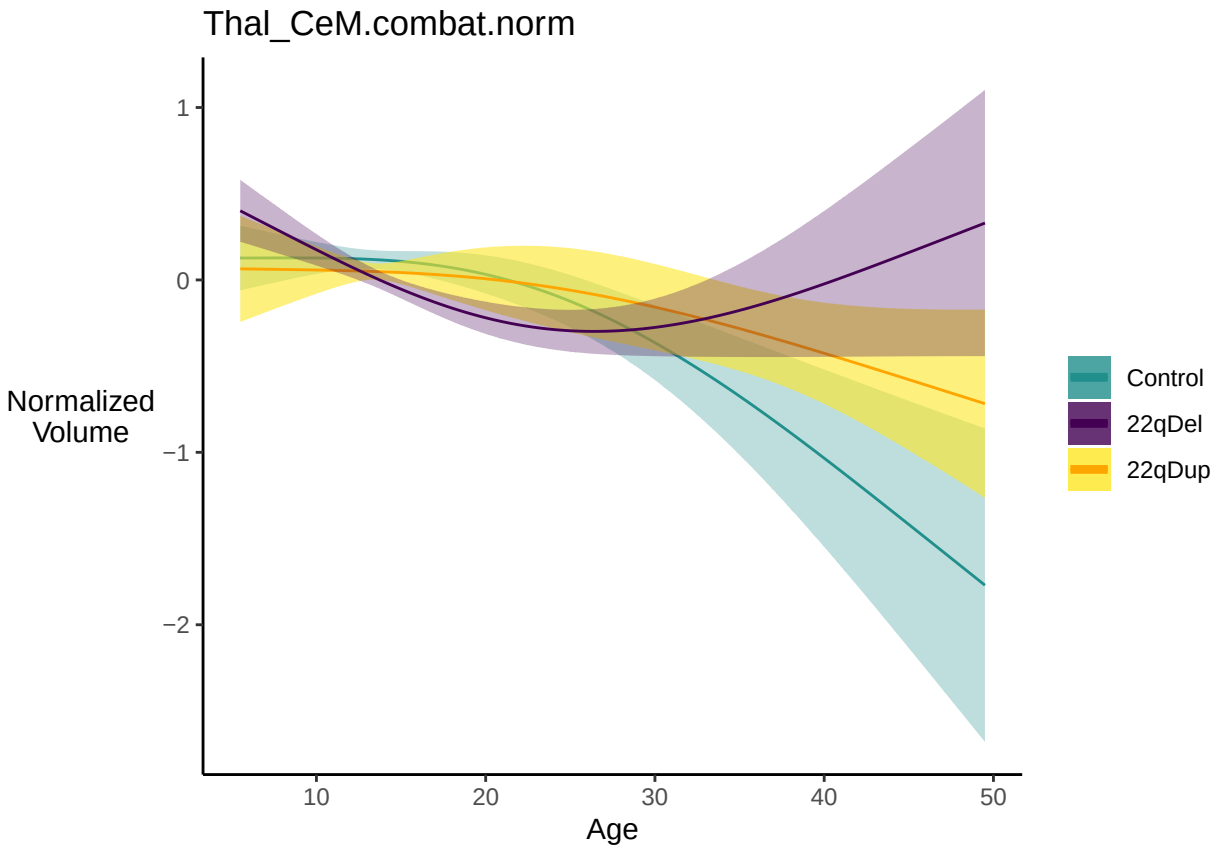

Hip_presubiculum.combat.norm
Normalized Volume
1.0
0.5
0.0
−0.5
−1.0
10
20
30
40
50
Age
Control
22qDel
22qDup

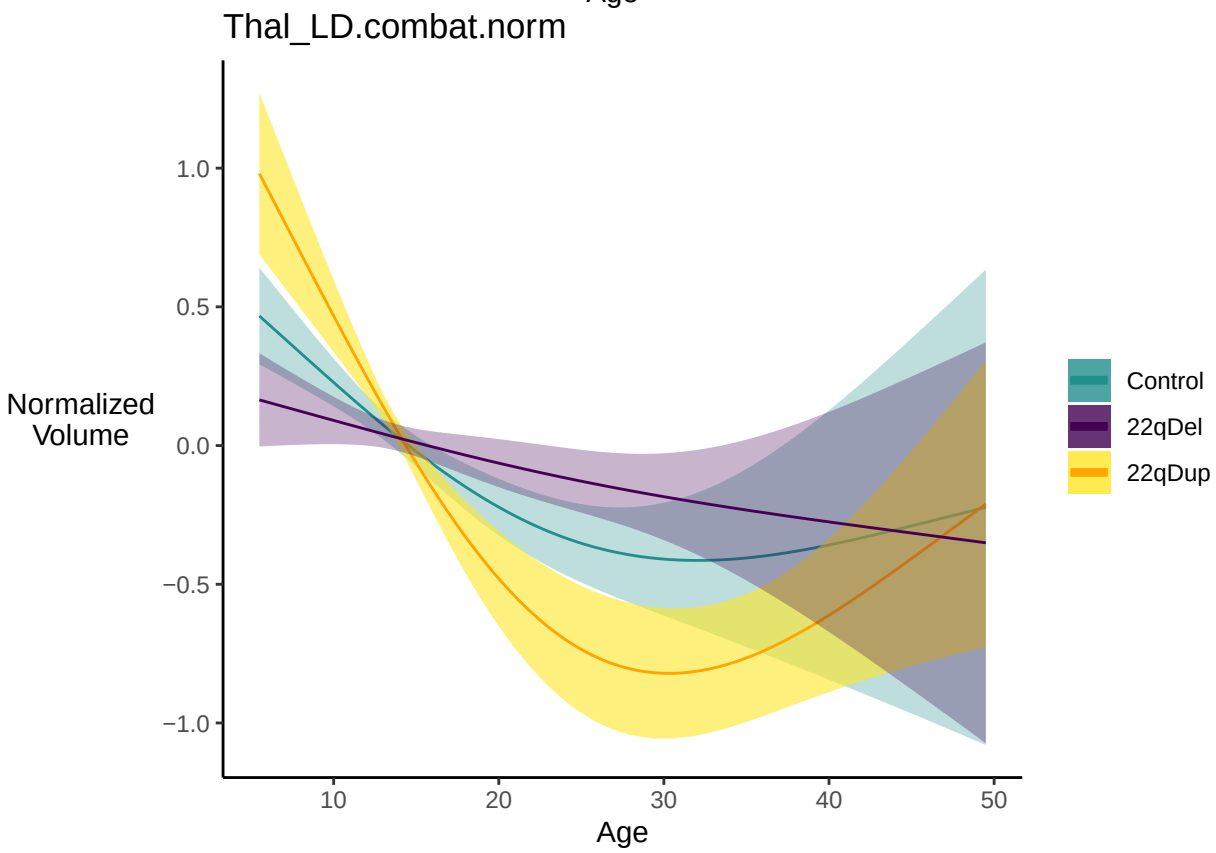

Hip_subiculum.combat.norm
Normalized Volume
0.5
0.0
−0.5
−1.0
10
20
30
40
50
Age
Control
22qDel
22qDup

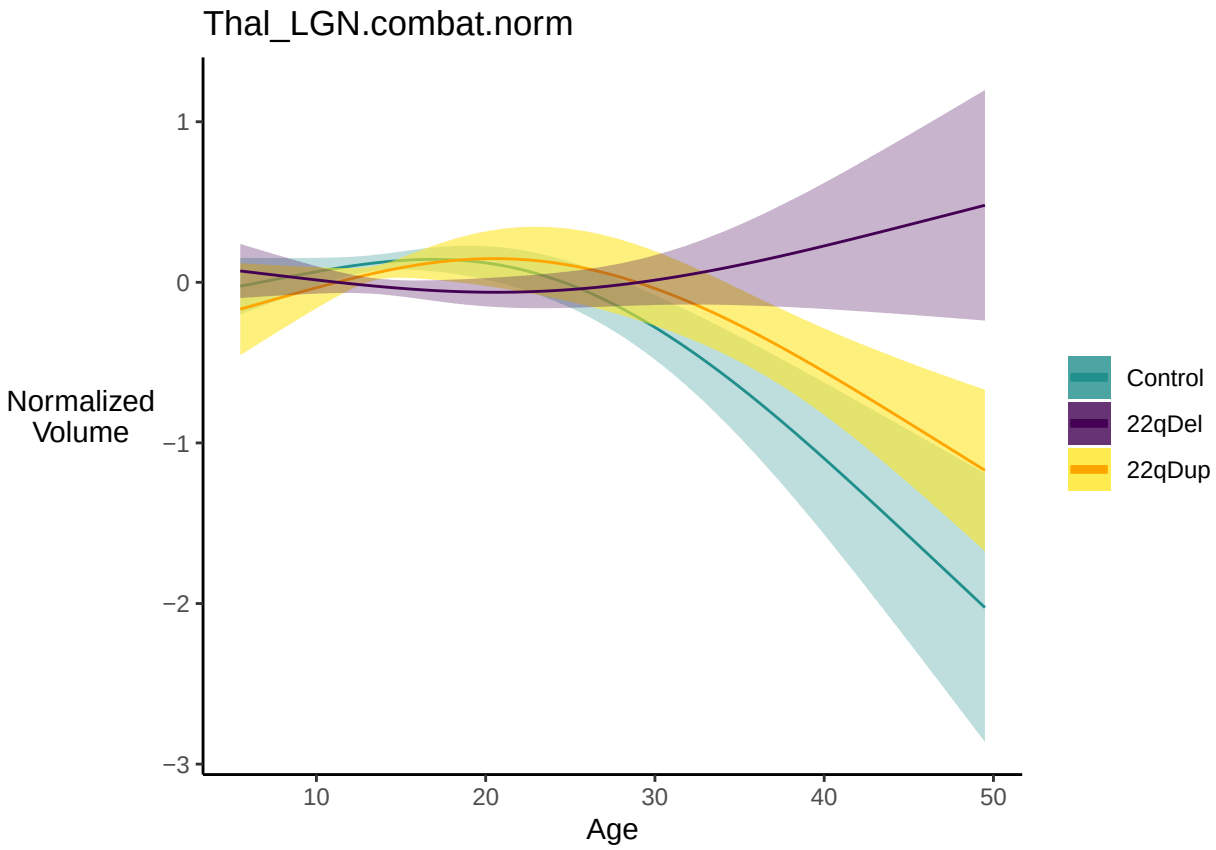

Thal_AV.combat.norm
Normalized Volume
1
0
−1
10
20
30
40
50
Age
Control
22qDel
22qDup

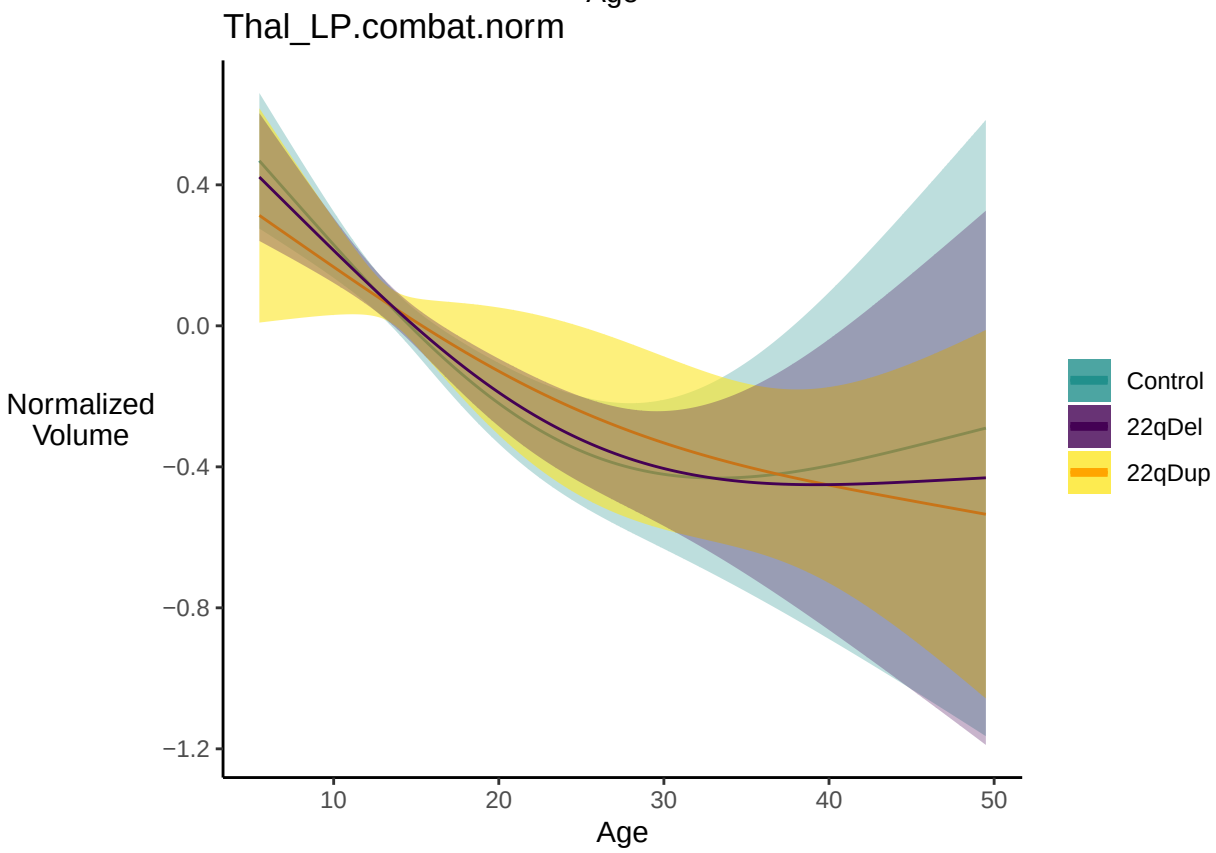

Thal_CL.combat.norm
Normalized Volume
1
0
−1
−2
10
20
30
40
50
Age
Control
22qDel
22qDup

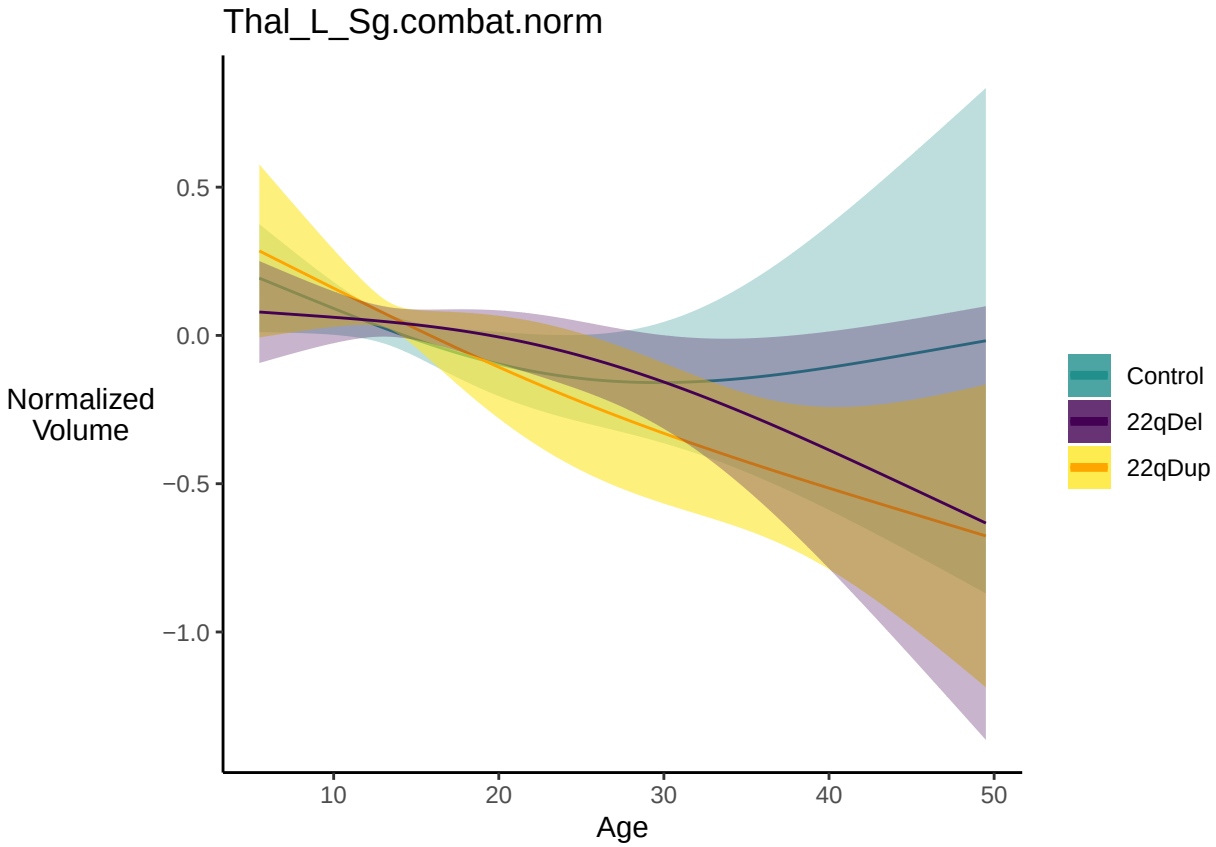

Thal_CM.combat.norm
Normalized Volume
1
0
−1
10
20
30
40
50
Age
Control
22qDel
22qDup

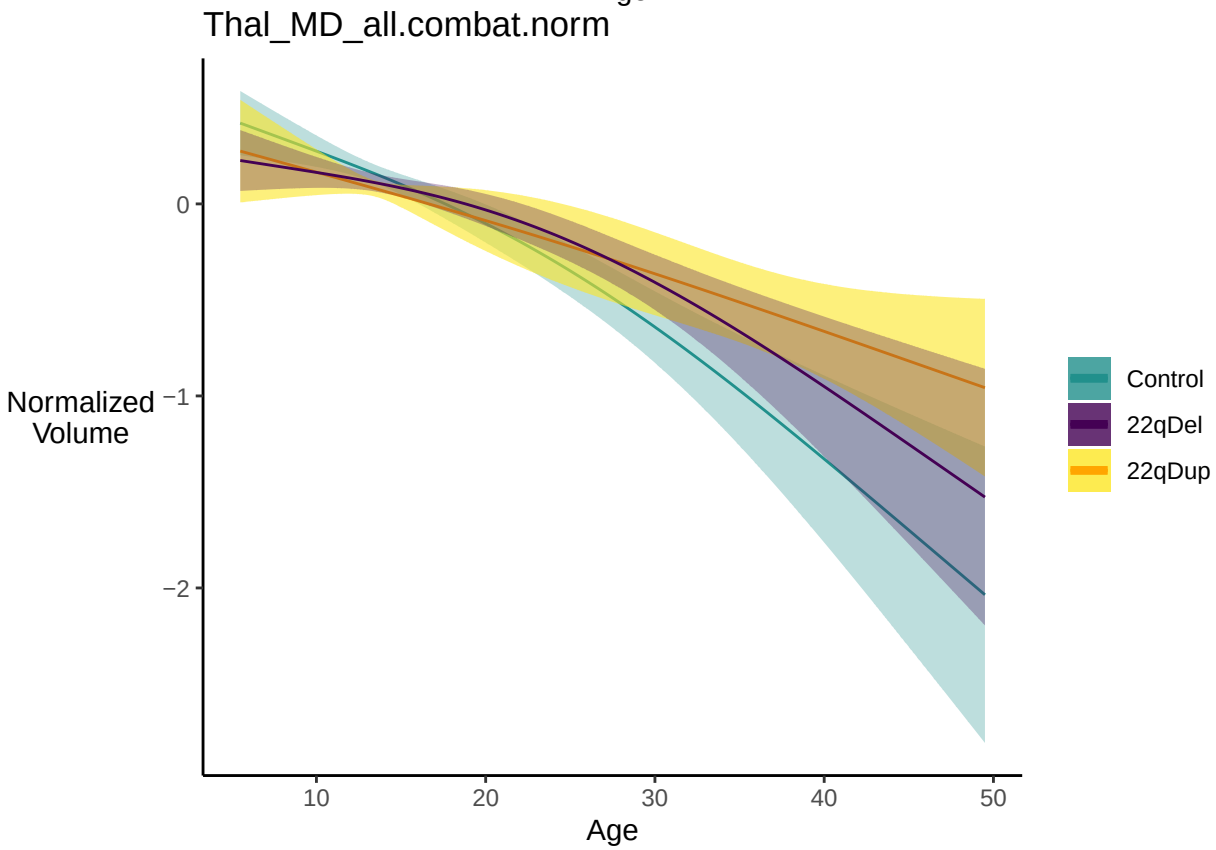

Thal_CeM.combat.norm
Normalized Volume
1
0
−1
−2
10
20
30
40
50
Age
Control
22qDel
22qDup

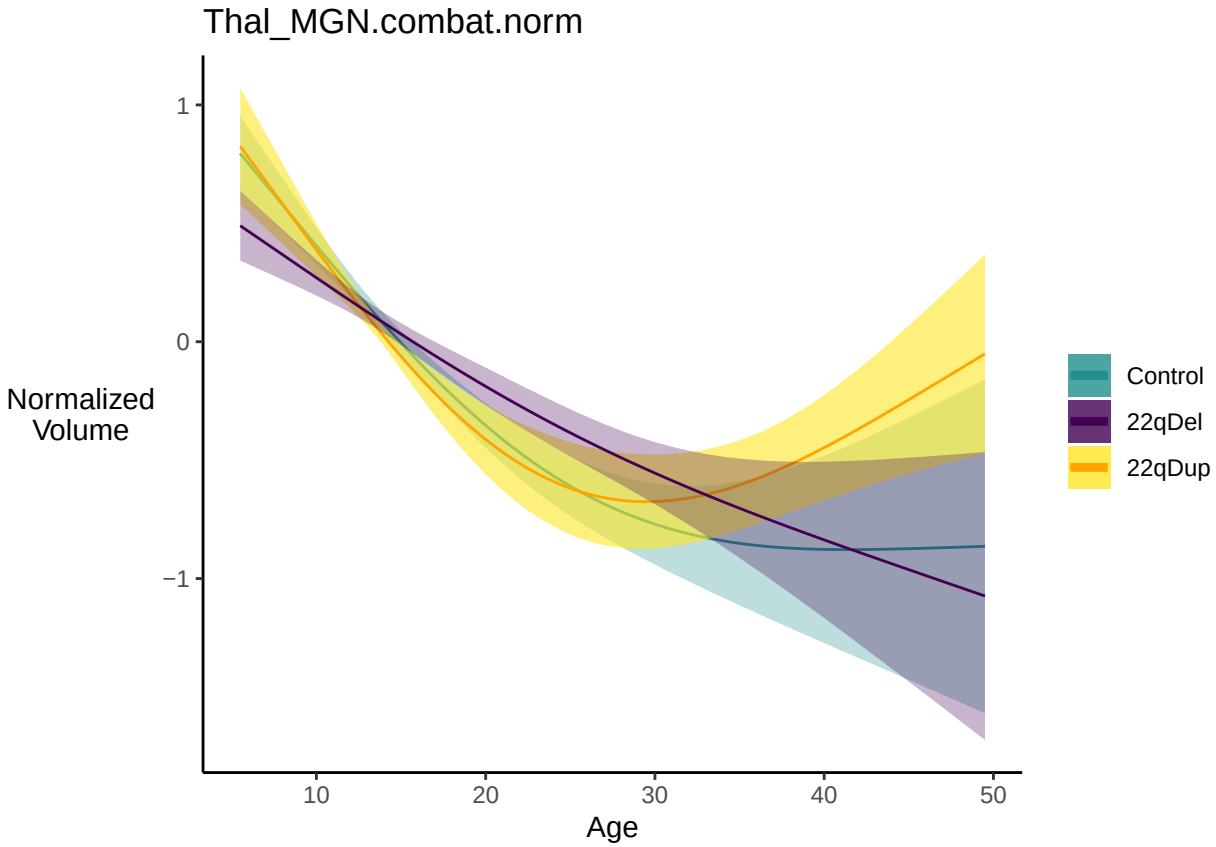

Thal_LD.combat.norm
Normalized Volume
1.0
0.5
0.0
−0.5
−1.0
10
20
30
40
50
Age
Control
22qDel
22qDup

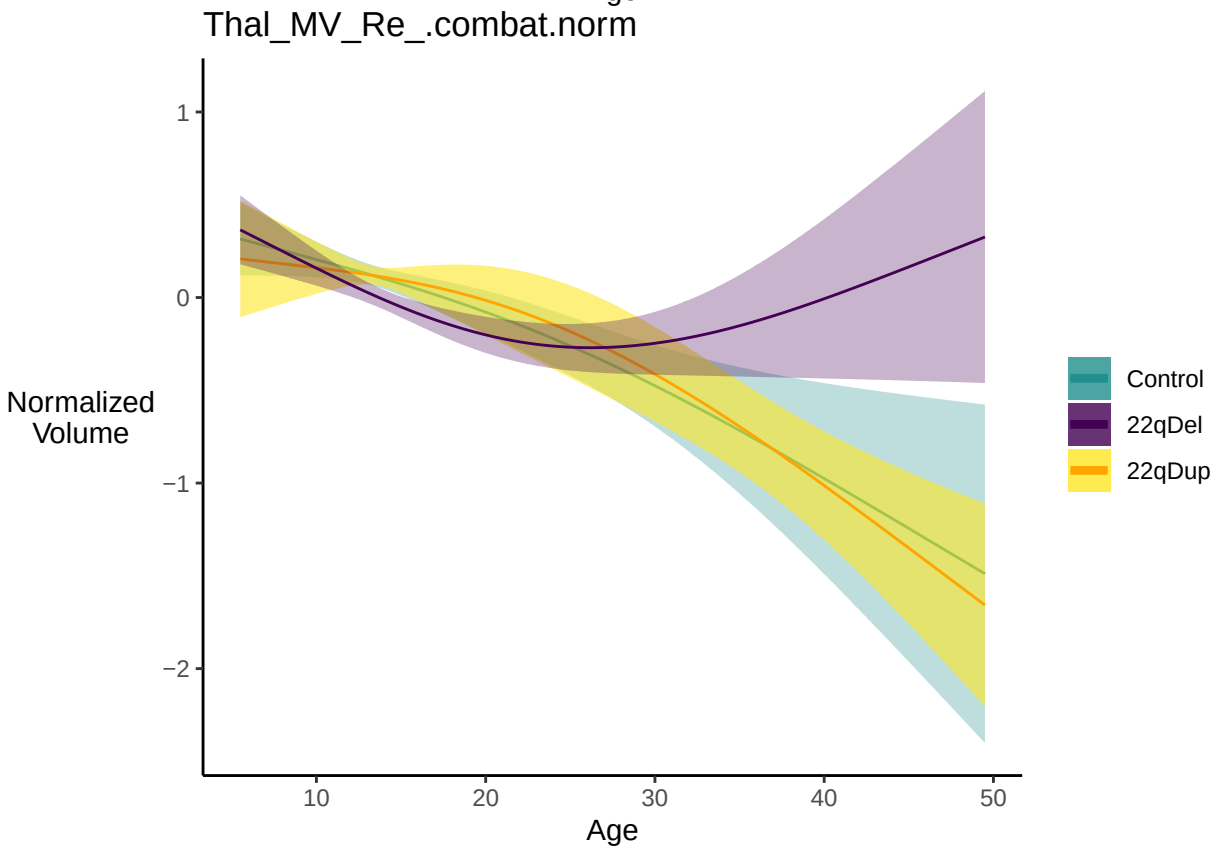

Thal_LGN.combat.norm
Normalized Volume
1
0
−1
−2
−3
10
20
30
40
50
Age
Control
22qDel
22qDup

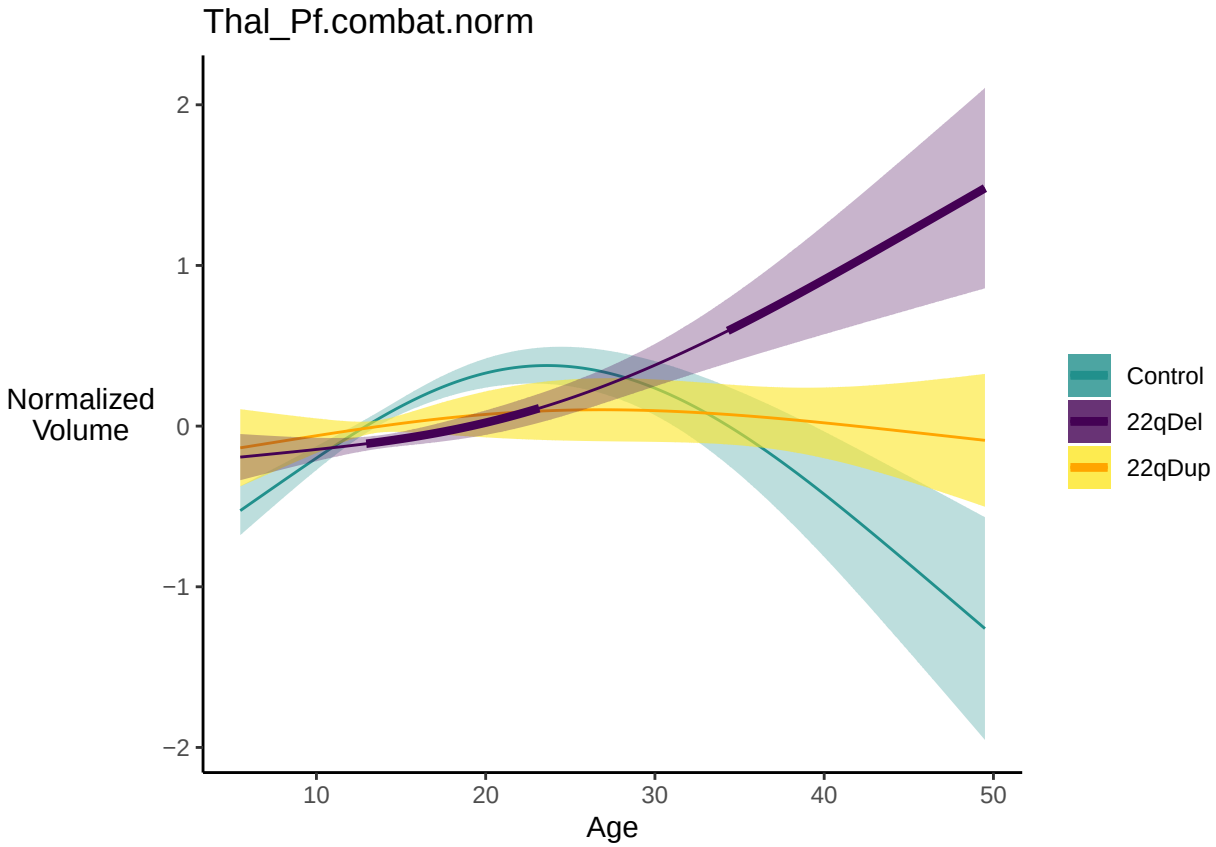

Thal_LP.combat.norm
Normalized Volume
0.4
0.0
−0.4
−0.8
−1.2
10
20
30
40
50
Age
Control
22qDel
22qDup

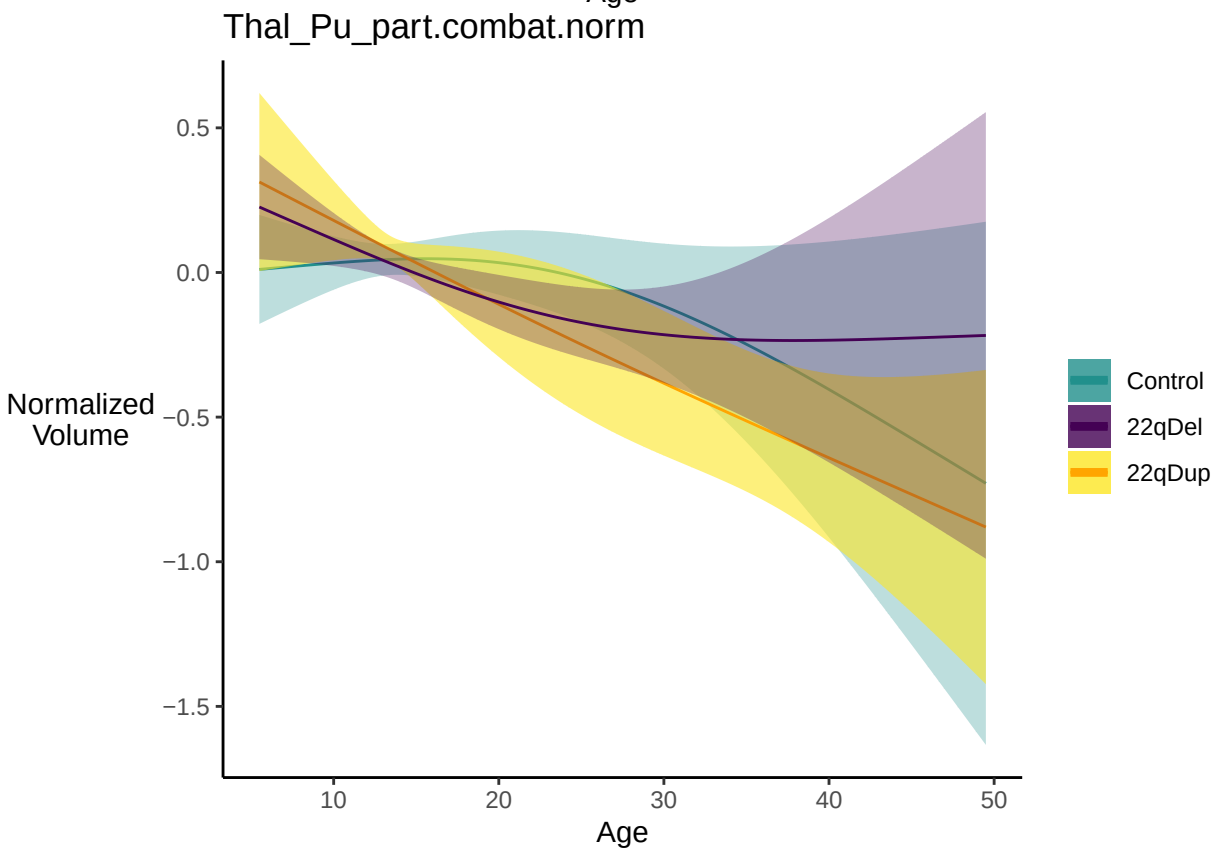

Thal_L_Sg.combat.norm
Normalized Volume
0.5
0.0
−0.5
−1.0
10
20
30
40
50
Age
Control
22qDel
22qDup

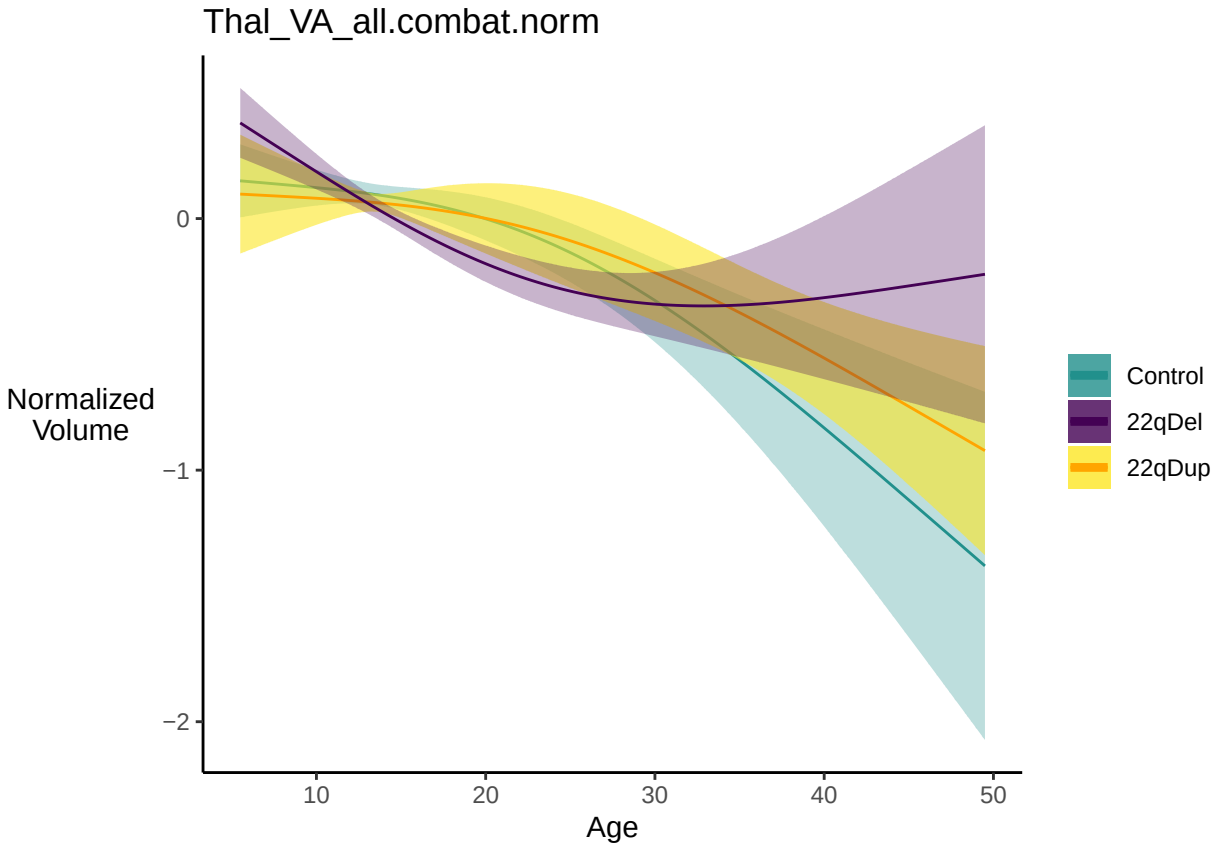

Thal_MD_all.combat.norm
Normalized Volume
0
−1
−2
10
20
30
40
50
Age
Control
22qDel
22qDup

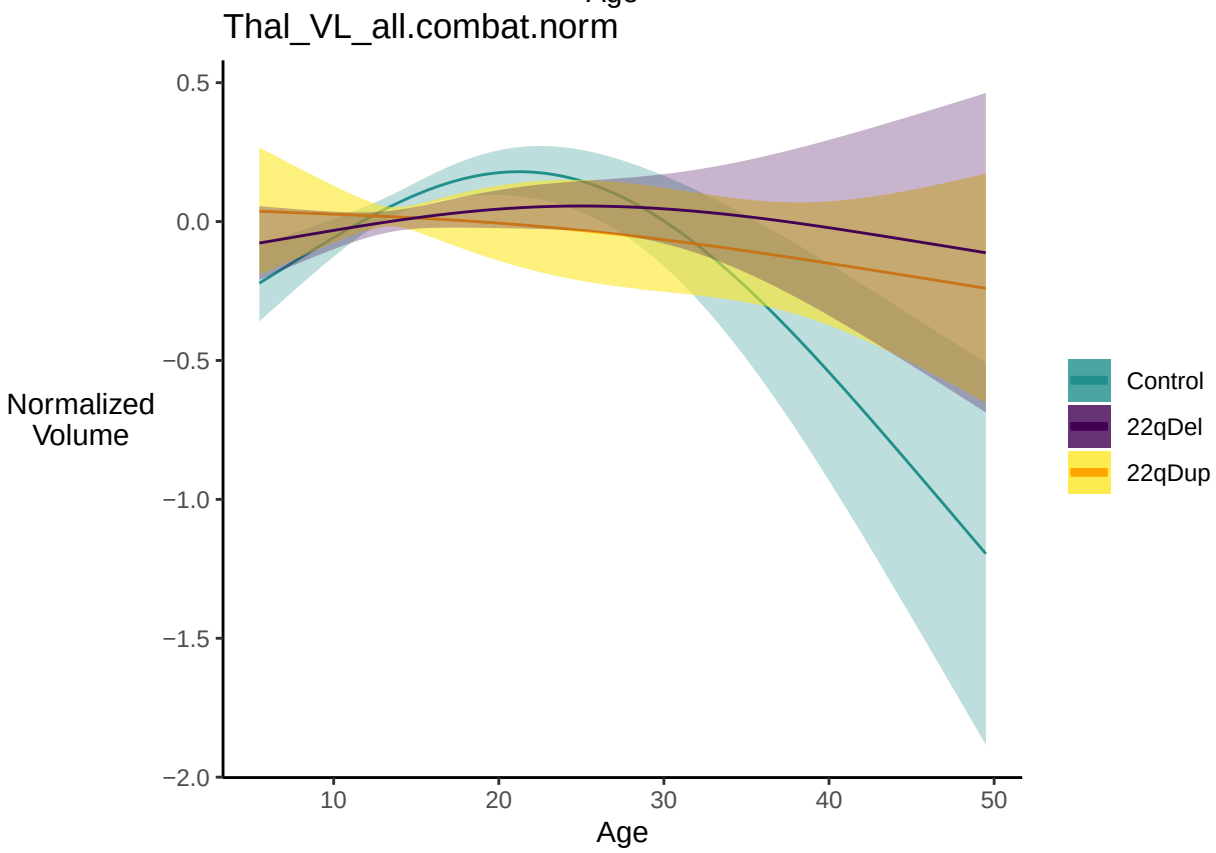

Thal_MGN.combat.norm
Normalized Volume
1
0
−1
10
20
30
40
50
Age
Control
22qDel
22qDup

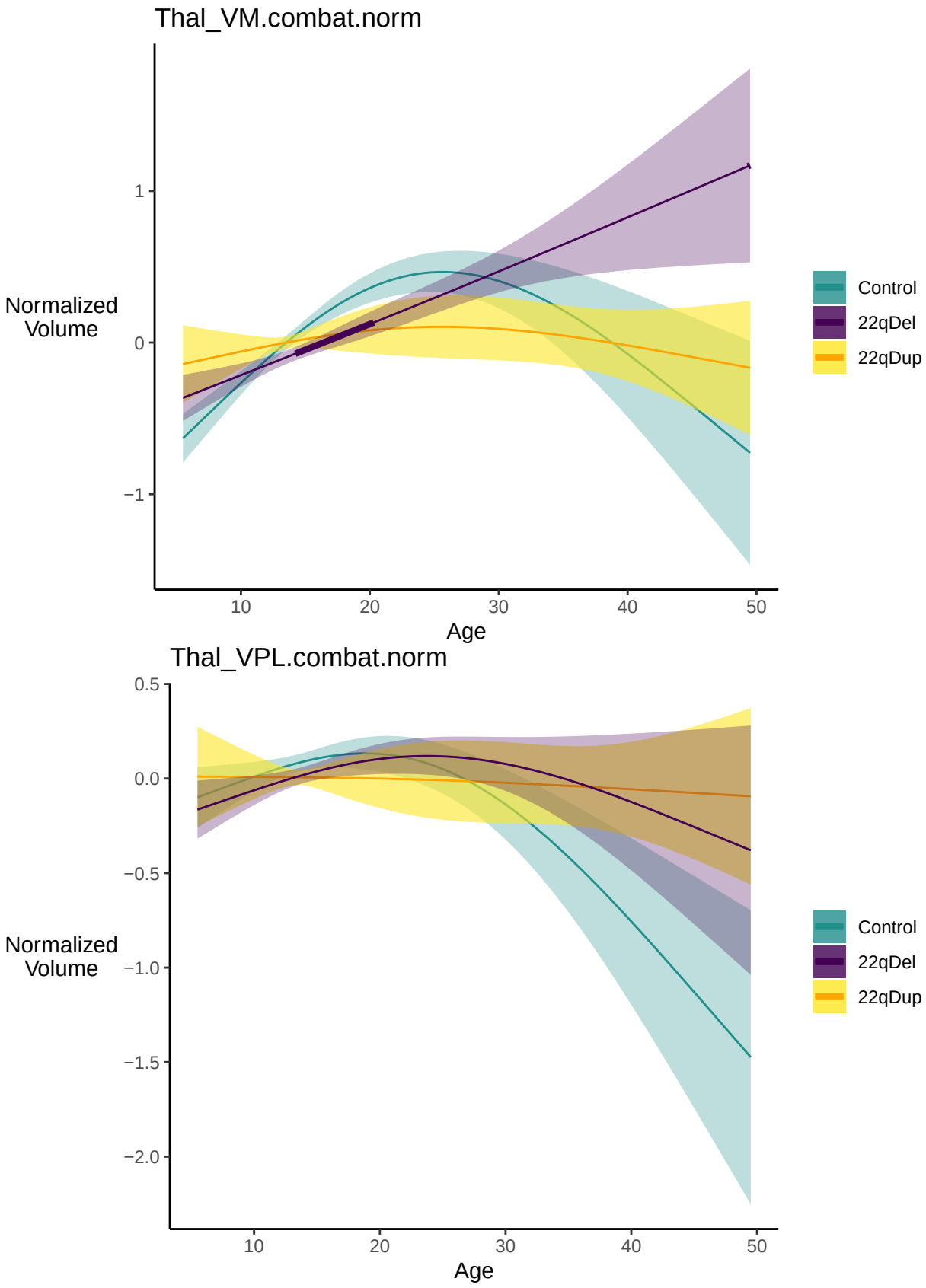

Thal_MV_Re_.combat.norm
Normalized Volume
1
0
−1
−2
10
20
30
40
50
Age
Control
22qDel
22qDup

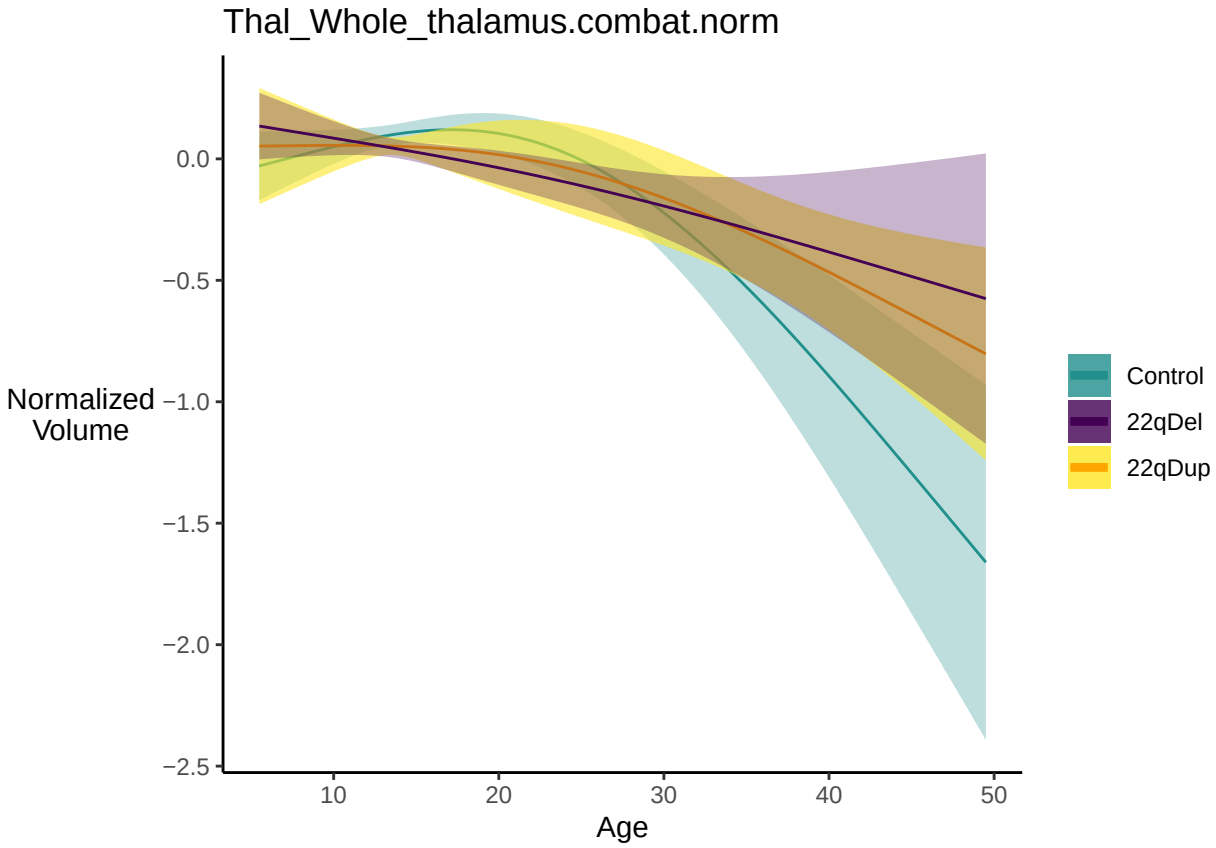

Thal_Pf.combat.norm
Normalized Volume
2
1
0
−1
−2
10
20
30
40
50
Age
Control
22qDel
22qDup

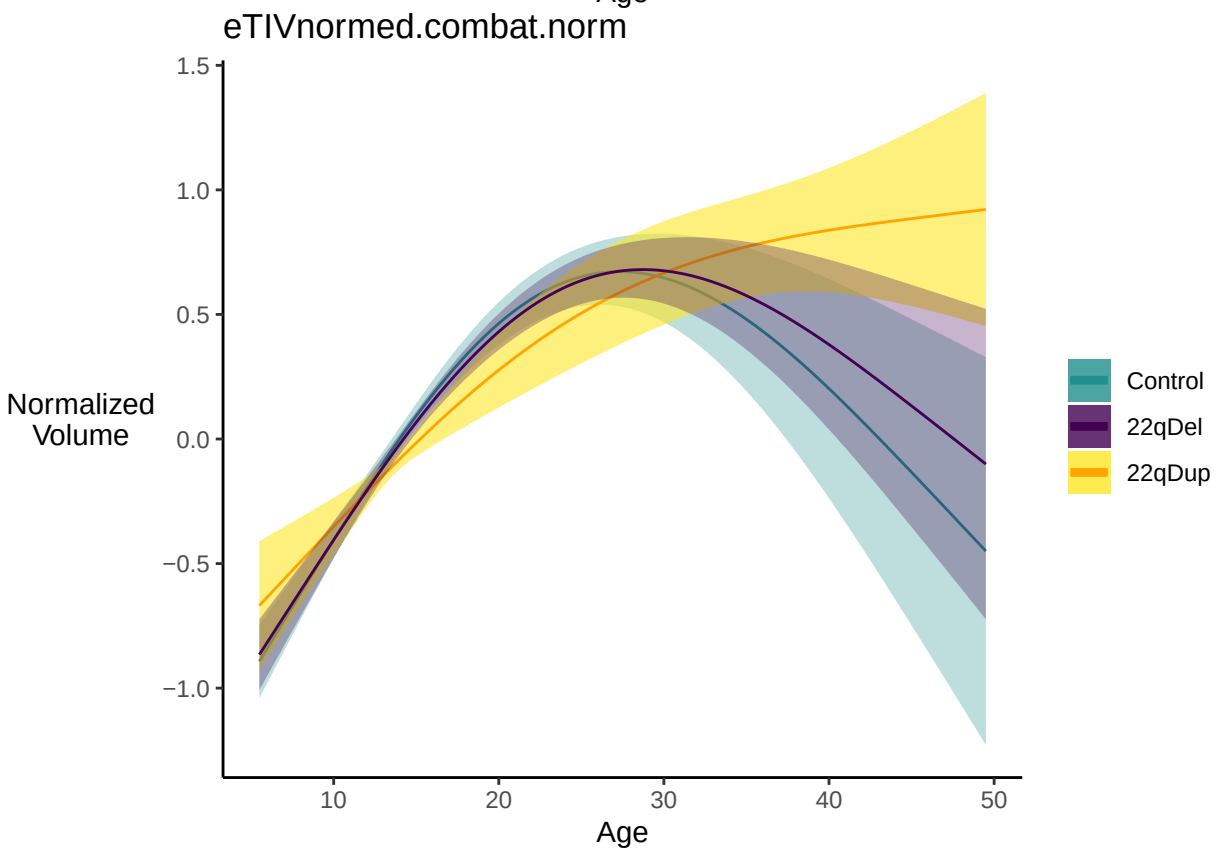

Thal_Pu_part.combat.norm
Normalized Volume
0.5
0.0
−0.5
−1.0
−1.5
10
20
30
40
50
Age
Control
22qDel
22qDup


Thal_VA_all.combat.norm
Normalized Volume
0
−1
−2
10
20
30
40
50
Age
Control
22qDel
22qDup


Thal_VL_all.combat.norm
Normalized Volume
0.5
0.0
−0.5
−1.0
−1.5
−2.0
10
20
30
40
50
Age
Control
22qDel
22qDup

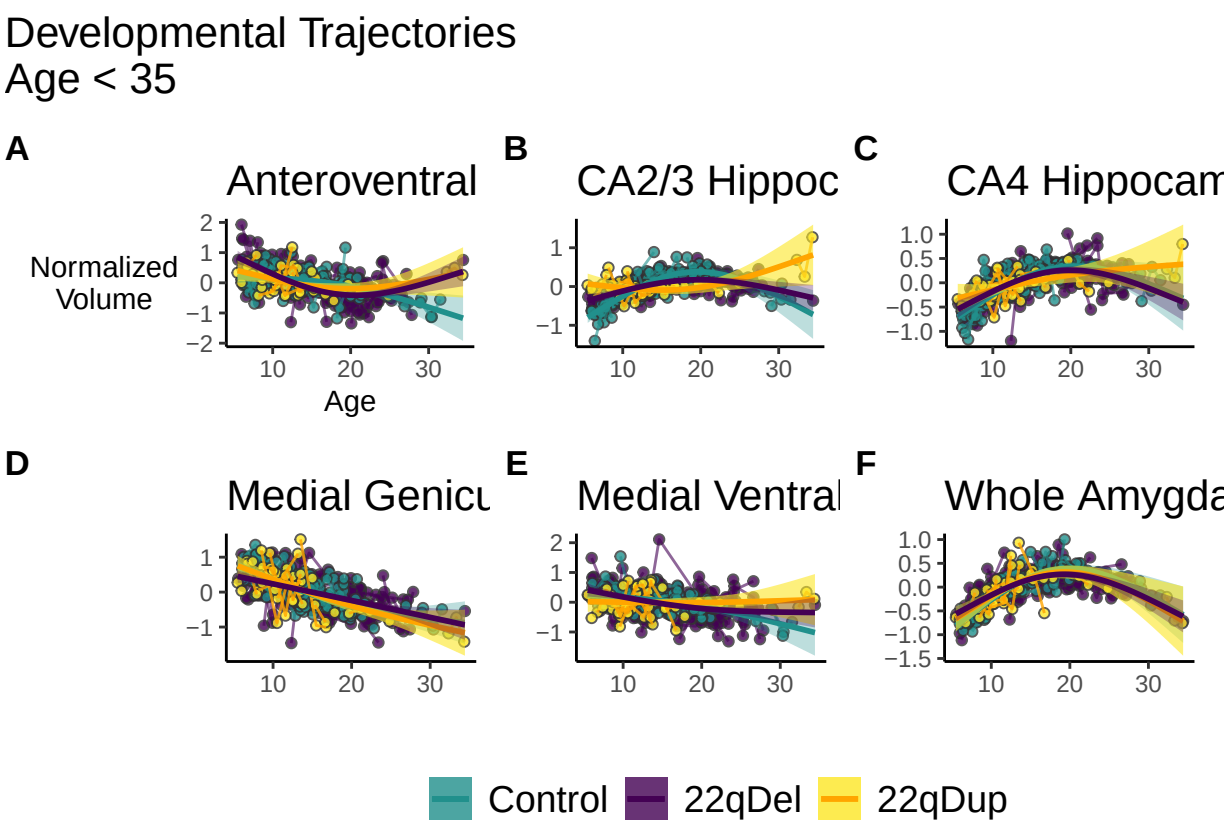

Thal_VM.combat.norm
Normalized Volume
1
0
−1
10
20
30
40
50
Age
Control
22qDel
22qDup
Thal_VPL.combat.norm
Normalized Volume
0.5
0.0
−0.5
−1.0
−1.5
−2.0
10
20
30
40
50
Age
Control
22qDel
22qDup





plot age gamms for young sample

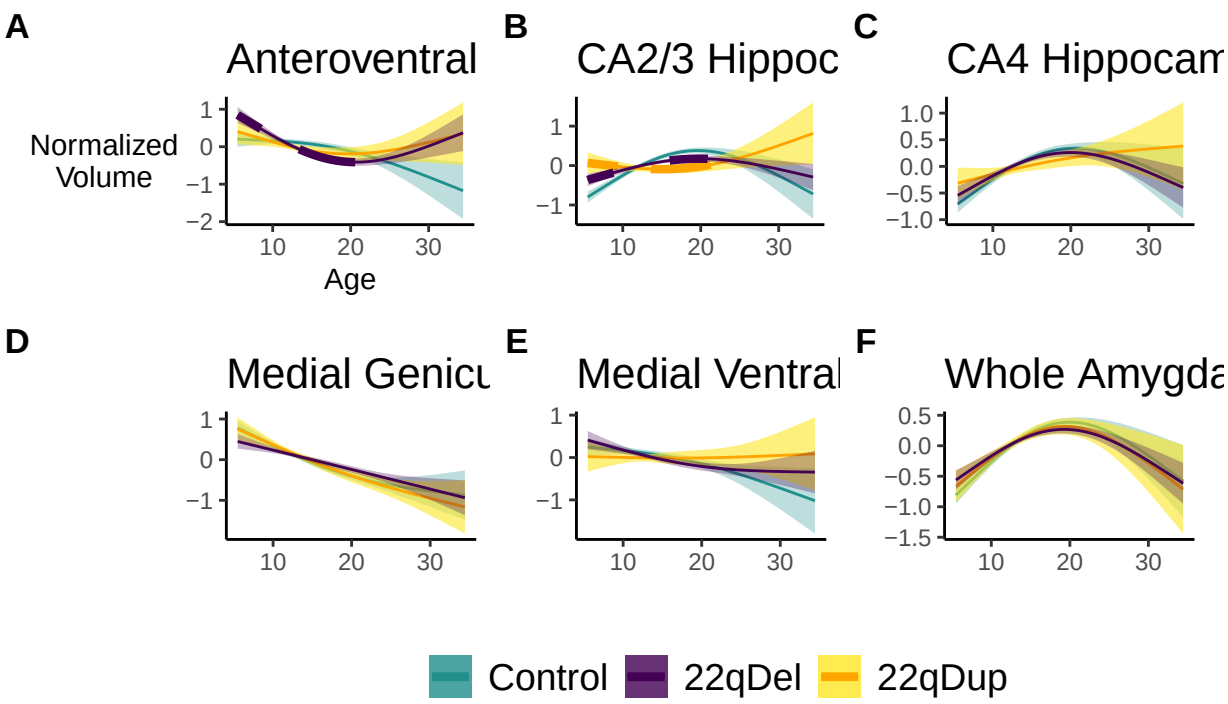
```
#plot chosen gamms
thal_av_young <- plot_gamm_resid(list=maturation_young, name =
 ↪  "Thal_AV.combat.norm", xlab= "Age", ylab= "Normalized\nVolume", title =
 ↪  "Anteroventral Thalamus")
hip_ca3_young <- plot_gamm_resid(list=maturation_young, name =
 ↪  "Hip_CA3.combat.norm", xlab= "", ylab= "", title = "CA2/3 Hippocampus")
amy_whole_young <- plot_gamm_resid(list=maturation_young, name =
 ↪  "Amy_Whole_amygdala.combat.norm", xlab= "", ylab= "", title = "Whole
 ↪  Amygdala")
thal_mgn_young <- plot_gamm_resid(list=maturation_young, name =
 ↪  "Thal_MGN.combat.norm", xlab= "", ylab= "", title = "Medial Geniculate
 ↪  Thalamus")
thal_mvre_young <- plot_gamm_resid(list=maturation_young, name =
 ↪  "Thal_MV_Re_.combat.norm", xlab= "", ylab= "", title = "Medial Ventral
 ↪  Thalamus")
hip_ca4_young <- plot_gamm_resid(list=maturation_young, name =
 ↪  "Hip_CA4.combat.norm", xlab= "", ylab= "", title = "CA4 Hippocampus")

plot_age_young <- (thal_av_young + hip_ca3_young + hip_ca4_young) /
 ↪  (thal_mgn_young + thal_mvre_young + amy_whole_young) + plot_layout(guides =
 ↪  "collect") + plot_annotation(tag_levels = 'A', title = "Developmental
 ↪  Trajectories\nAge < 35") &   theme(plot.tag = element_text(face = 'bold')) +
 ↪  theme(legend.position = "bottom", legend.text=element_text(size = 14),
 ↪  plot.title = element_text(size = 16, hjust = 0))

plot_age_young
```



```
#ggsave(plot=plot_age_young, filename =file.path(project,
 ↪  "figures/age/age_gamms_curves_resid_u35.pdf"), width= 12, height= 7, device =
 ↪  "pdf")
#ggsave(plot=plot_age_young, filename =file.path(project,
 ↪  "figures/age/age_gamms_curves_resid_u35.png"), width= 12, height= 7, device =
 ↪  "png", dpi = 300)
```

plot age gamms with group differences for young sample

```
#plot chosen gamms
thal_av_young_gd <- plot_gamm_gdiff(list=mat_young_gdiff, name =
 ↪  "Thal_AV.combat.norm", xlab= "Age", ylab= "Normalized\nVolume", title =
 ↪  "Anteroventral Thalamus")
hip_ca3_young_gd <- plot_gamm_gdiff(list=mat_young_gdiff, name =
 ↪  "Hip_CA3.combat.norm", xlab= "", ylab= "", title = "CA2/3 Hippocampus")
amy_whole_young_gd <- plot_gamm_gdiff(list=mat_young_gdiff, name =
 ↪  "Amy_Whole_amygdala.combat.norm", xlab= "", ylab= "", title = "Whole
 ↪  Amygdala")
thal_mgn_young_gd <- plot_gamm_gdiff(list=mat_young_gdiff, name =
 ↪  "Thal_MGN.combat.norm", xlab= "", ylab= "", title = "Medial Geniculate
 ↪  Thalamus")
thal_mvre_young_gd <- plot_gamm_gdiff(list=mat_young_gdiff, name =
 ↪  "Thal_MV_Re_.combat.norm", xlab= "", ylab= "", title = "Medial Ventral
 ↪  Thalamus")
hip_ca4_young_gd <- plot_gamm_gdiff(list=mat_young_gdiff, name =
 ↪  "Hip_CA4.combat.norm", xlab= "", ylab= "", title = "CA4 Hippocampus")

plot_age_young_gd <- (thal_av_young_gd + hip_ca3_young_gd + hip_ca4_young_gd) /
 ↪  (thal_mgn_young_gd + thal_mvre_young_gd + amy_whole_young_gd) +
 ↪  plot_layout(guides = "collect") + plot_annotation(tag_levels = 'A', title =
 ↪  "Developmental Trajectories\nAge < 35") &   theme(plot.tag = element_text(face
 ↪  = 'bold')) + theme(legend.position = "bottom", legend.text=element_text(size =
 ↪  14), plot.title = element_text(size = 16, hjust = 0))

plot_age_young_gd
```




```
#ggsave(plot=plot_age_young_gd, filename =file.path(project,
 ↪  "figures/age/age_gamms_curves_gdiff_u35.pdf"), width= 12, height= 7, device =
 ↪  "pdf")
#ggsave(plot=plot_age_young_gd, filename =file.path(project,
 ↪  "figures/age/age_gamms_curves_gdiff_u35.png"), width= 12, height= 7, device =
 ↪  "png", dpi = 300)
```

supplemental

export table of roi sizes

```
# get average roi per group
group_mean_vols <- data.frame(Region=sc_names)
for(c in sc_names){
  group_mean_vols[which(group_mean_vols$Region == c),"22qDel"] <-
 ↪  mean(filter(demo_sc, SUBJECT_IDENTITY== "PATIENT-DEL")[, c], na.rm = TRUE) %>%
 ↪  round(., digits = 0)
  group_mean_vols[which(group_mean_vols$Region == c),"TD"] <- mean(filter(demo_sc,
 ↪  SUBJECT_IDENTITY== "CONTROL")[, c], na.rm = TRUE) %>% round(., digits = 0)
  group_mean_vols[which(group_mean_vols$Region == c),"22qDup"] <-
 ↪  mean(filter(demo_sc, SUBJECT_IDENTITY== "PATIENT-DUP")[, c], na.rm = TRUE) %>%
 ↪  round(., digits = 0)
}
group_mean_vols <- group_mean_vols[order(group_mean_vols$TD),]

# get region names for matching
group_mean_vols$r <- gsub("Thal_", "", group_mean_vols$Region)
group_mean_vols$r <- gsub("Amy_", "", group_mean_vols$r)
```

```
group_mean_vols$r <- gsub("Hip_", "", group_mean_vols$r)

# merge with full names
#save_dosage_effect <- merge(x =save_dosage_effect, y =lut[c("Structure", "Name",
 ↪  "region_match")], by.x = "Region", by.y = "region_match", all.x =TRUE)
group_mean_match <- merge(x =group_mean_vols, y =lut_unique[, c("Structure",
 ↪  "bilat_name", "bilat_match")], by.x = "r", by.y = "bilat_match", all.x =TRUE)

# select cols
group_mean_out <- group_mean_match[order(group_mean_match[,"22qDel"]),
 ↪  c("Structure", "bilat_name", "22qDel", "TD", "22qDup")] %>% rename("Region"=
 ↪  "bilat_name")

group_mean_table <- group_mean_out %>% gt() %>%
  tab_style(style = cell_text(weight = "bold"), locations = list(cells_title(),
   ↪  cells_column_labels())) %>%
  cols_align(align= "right", columns =everything()) %>%
  tab_header("Average volumes (mm^3) by group")%>%
  tab_options(data_row.padding = px(1))
group_mean_table
# gtsave(group_mean_table, filename = file.path(project,
 ↪  "figures/atlas/volumes.pdf"))
# gtsave(group_mean_table, filename = file.path(project,
 ↪  "figures/atlas/volumes.png"))
# gtsave(group_mean_table, filename = file.path(project,
 ↪  "figures/atlas/volumes.rtf"))

# list in order for methods
group_mean_write <- group_mean_out
setorder(group_mean_write,Structure,Region)
```

To test for group differences, not just gene dosage effects (e.g. del < TD & Dup < TD or Del < TD = Dup), use a model with a binary variable for each group? or two separate case/control models?

```
#gamm_combat <- lapply(sc_names_normed, function(r) gam(formula = reformulate(
 ↪  c("s(AGE, by =SUBJECT_IDENTITY, bs =\"tp\", k= 3, fx =TRUE)", "gene_dosage",
 ↪  "SEX", "eTIVnormed", "site", "s(SUBJECTID, bs =\"re\", k= 3)"), response = r),
 ↪  data =demo_combat_na_gam, selection=TRUE, method = "REML", na.action=
 ↪  "na.omit"))

# test random intercept model at each region separately in Del and Dup
gam_combat_del <- lapply(sc_names_normed, function(r) gam(formula = reformulate(
 ↪  c("s(AGE, by =SUBJECT_IDENTITY, bs =\"tp\", k= 3, fx =TRUE)",
 ↪  "SUBJECT_IDENTITY", "SEX", "eTIVnormed", "site", "s(SUBJECTID, bs =\"re\", k=
 ↪  3)"), response = r), data =filter(demo_combat_na_gam, SUBJECT_IDENTITY %in%
 ↪  c("CONTROL", "PATIENT-DEL")), REML=TRUE))

gam_combat_dup <- lapply(sc_names_normed, function(r) gam(formula = reformulate(
 ↪  c("s(AGE, by =SUBJECT_IDENTITY, bs =\"tp\", k= 3, fx =TRUE)",
 ↪  "SUBJECT_IDENTITY", "SEX", "eTIVnormed", "site", "s(SUBJECTID, bs =\"re\", k=
 ↪  3)"), response = r), data =filter(demo_combat_na_gam,  SUBJECT_IDENTITY %in%
 ↪  c("CONTROL", "PATIENT-DUP")), REML=TRUE))

# get p-val for gene dosage effect
```

```
gene_dosage_effect_del <- lapply(gam_combat_del, function(l)
 ↪  summary(l)$p.table["SUBJECT_IDENTITYPATIENT-DEL", c("Estimate", "Pr(>|t|)")])
 ↪  %>% do.call(rbind,.) %>% as.data.frame
colnames(gene_dosage_effect_del) <- c("gene_dosage_beta", "gene_dosage_p")
gene_dosage_effect_del$Region <- sc_names_normed
gene_dosage_effect_del$model <- "Del"

gene_dosage_effect_dup <- lapply(gam_combat_dup, function(l)
 ↪  summary(l)$p.table["SUBJECT_IDENTITYPATIENT-DUP", c("Estimate", "Pr(>|t|)")])
 ↪  %>% do.call(rbind,.) %>% as.data.frame
colnames(gene_dosage_effect_dup) <- c("gene_dosage_beta", "gene_dosage_p")
gene_dosage_effect_dup$Region <- sc_names_normed
gene_dosage_effect_dup$model <- "Dup"

del_dup_vs_hc <- rbind(gene_dosage_effect_del, gene_dosage_effect_dup)
del_dup_vs_hc$gene_dosage_fdr_q <- p.adjust(del_dup_vs_hc$gene_dosage_p, method =
 ↪  "fdr")
del_dup_vs_hc$fdr_sig <- del_dup_vs_hc$gene_dosage_fdr_q < 0.05

del_dup_vs_hc_merged <- merge(x =filter(del_dup_vs_hc, model == "Del"), y
 ↪  =filter(del_dup_vs_hc, model == "Dup"), by = "Region")

# number of sig 22qDel vs control
sum(del_dup_vs_hc_merged$fdr_sig.x)

# number of sig 22qDup vs control
sum(del_dup_vs_hc_merged$fdr_sig.y)

# get only effects that are significant in both models
del_dup_vs_hc_merged_sig <- filter(del_dup_vs_hc_merged, fdr_sig.x ==TRUE &
 ↪  fdr_sig.y ==TRUE)
#del_dup_vs_hc_sig <-
 ↪  del_dup_vs_hc_sig[order(del_dup_vs_hc_sig$gene_dosage_beta),]
```

export results of individual group comparisons

```
group_compare_sort <- del_dup_vs_hc
group_compare_sort$Group <- group_compare_sort$model %>% gsub("Del", "22qDel",.)
 ↪  %>% gsub("Dup", "22qDup",.)

# get region names for matching
group_compare_sort$Region <- gsub(".combat.norm", "", group_compare_sort$Region)
group_compare_sort$Region <- gsub("Thal_", "", group_compare_sort$Region)
group_compare_sort$Region <- gsub("Amy_", "", group_compare_sort$Region)
group_compare_sort$Region <- gsub("Hip_", "", group_compare_sort$Region)

# merge with full names
group_compare_sort <- merge(x =group_compare_sort, y =lut_unique[, c("Structure",
 ↪  "bilat_name", "bilat_match")], by.x = "Region", by.y = "bilat_match", all.x
 ↪  =TRUE)

# set order
group_compare_sort$struct_order <- group_compare_sort$Structure %>%
 ↪  gsub("thalamus",1,.) %>% gsub("hippocampus",2,.) %>% gsub("amygdala",3,.)
```

```
setorder(group_compare_sort, model, struct_order, gene_dosage_p)

# round
group_compare_sort$gene_dosage_beta %<>% round(., digits = 2) %>%
 ↪  sprintf("%.2f",.)
#save_dosage_effect$p %<>% round(., digits = 3) %>% sprintf("%.3f",.)
group_compare_sort$gene_dosage_p %<>% formatC(., format = "g", digits = 2)
#save_dosage_effect$`FDR q` %<>% round(., digits = 6) %>% sprintf("%.6f",.)
# g option formats as scientific only when saves space
group_compare_sort$gene_dosage_fdr_q %<>% formatC(., format = "g", digits = 2)

group_compare_out <- data.frame(Group=group_compare_sort$Group,
                                Structure =group_compare_sort$Structure,
                                Region=group_compare_sort$bilat_name,
                                beta =group_compare_sort$gene_dosage_beta,
                                p=group_compare_sort$gene_dosage_p,
                                FDR_q=group_compare_sort$gene_dosage_fdr_q)

# save groups separately
group_compare_del <- filter(group_compare_out, Group== "22qDel")
group_compare_dup <- filter(group_compare_out, Group== "22qDup")

# edit structure names
structure_dup <- duplicated(group_compare_del$Structure)
for(i in 1:length(structure_dup)){
  if(structure_dup[i]==TRUE){
    group_compare_del[i,"Structure"] <- ""
  }
}

structure_dup <- duplicated(group_compare_dup$Structure)
for(i in 1:length(structure_dup)){
  if(structure_dup[i]==TRUE){
    group_compare_dup[i,"Structure"] <- ""
  }
}

#write.csv(group_compare_del[, c("Structure", "Region", "beta", "p", "FDR_q")],
 ↪  file =file.path(project, "figures/gene_dosage/del_vs_td.csv"), row.names =
 ↪  FALSE)
#write.csv(group_compare_dup[, c("Structure", "Region", "beta", "p", "FDR_q")],
 ↪  file =file.path(project, "figures/gene_dosage/dup_vs_td.csv"), row.names =
 ↪  FALSE)

# export table
group_compare_del_out <- group_compare_del[, c("Structure", "Region", "beta", "p",
 ↪  "FDR_q")] %>% gt() %>%
  tab_style(style = cell_text(weight = "bold"), locations =
   ↪  list(cells_column_labels())) %>%
  cols_align(align= "right", columns =everything()) %>%
  tab_options(data_row.padding = px(1))
group_compare_del_out
```

Structure	Region	beta	p	FDR_q
thalamus	lateral geniculate	-0.73	5.2e-07	1.4e-05
	medial ventral (reuniens)	-0.60	6.7e-05	0.00092
	medial geniculate	-0.31	0.008	0.044
	parafascicular	-0.34	0.01	0.051
	mediodorsal	0.30	0.022	0.095
	lateral posterior	-0.25	0.11	0.33
	anteroventral	-0.30	0.12	0.35
	central medial	-0.24	0.15	0.42
	limitans (suprageniculate)	0.19	0.19	0.48
	ventral lateral	0.21	0.26	0.54
	laterodorsal	0.15	0.29	0.57
	ventral anterior	-0.14	0.41	0.67
	whole thalamus	-0.15	0.5	0.76
	pulvinar	-0.12	0.51	0.76
	centromedian	-0.05	0.75	0.91
	ventral posterolateral	-0.04	0.78	0.91
	ventromedial	-0.03	0.84	0.93
	central lateral	-0.02	0.87	0.93
hippocampus	hippocampal tail	-1.01	1.6e-07	1.3e-05
	subiculum	-0.94	3.5e-07	1.4e-05
	molecular layer	-0.91	1.6e-05	0.00032
	whole hippocampus	-0.85	4.2e-05	0.00069
	hippocampal fissure	-0.61	8.7e-05	0.00093
	GC ML DG	-0.78	9.1e-05	0.00093
	CA4	-0.75	0.00012	0.0011
	CA1	-0.77	0.00023	0.0019
	presubiculum	-0.25	0.14	0.41
	CA2/3	-0.22	0.18	0.48
	parasubiculum	0.21	0.2	0.49
	fimbria	-0.14	0.32	0.59
	hippocampal amygdala transition area	0.17	0.33	0.59
amygdala	basal nucleus	-0.54	0.0011	0.008
	medial nucleus	-0.46	0.0012	0.008
	paralaminar nucleus	-0.48	0.0014	0.0086
	accessory basal nucleus	-0.39	0.013	0.061
	cortical nucleus	-0.33	0.022	0.095
	whole amygdala	-0.36	0.027	0.11
	central nucleus	-0.30	0.043	0.16
	lateral nucleus	-0.17	0.23	0.51
	anterior amygdaloid area	-0.12	0.36	0.62
	corticoamygdaloid transition	0.11	0.51	0.76

```
# gtsave(group_compare_del_out, filename = file.path(project,
 ↪  "figures/gene_dosage/del_vs_td_volumes.png"))
# gtsave(group_compare_del_out, filename = file.path(project,
 ↪  "figures/gene_dosage/del_vs_td_volumes.pdf"))
# gtsave(group_compare_del_out, filename = file.path(project,
 ↪  "figures/gene_dosage/del_vs_td_volumes.rtf"))
# gtsave(group_compare_del_out, filename = file.path(project,
 ↪  "figures/gene_dosage/gene_dosage_table.tex"))
```

```
# export table
group_compare_dup_out <- group_compare_dup[, c("Structure", "Region", "beta", "p",
 ↪  "FDR_q")] %>% gt() %>%
  tab_style(style = cell_text(weight = "bold"), locations =
   ↪  list(cells_column_labels())) %>%
  cols_align(align= "right", columns =everything()) %>%
  tab_options(data_row.padding = px(1))
group_compare_dup_out
```

Structure	Region	beta	p	FDR_q
thalamus	mediodorsal	-0.44	0.0077	0.044
	ventral posterolateral	-0.32	0.087	0.28
	limitans (suprageniculate)	0.31	0.11	0.33
	ventromedial	-0.24	0.2	0.49
	laterodorsal	0.24	0.22	0.51
	lateral posterior	0.23	0.24	0.53
	pulvinar	0.26	0.27	0.56
	central lateral	0.22	0.33	0.59
	ventral lateral	-0.17	0.38	0.65
	lateral geniculate	-0.16	0.41	0.67
	medial geniculate	-0.11	0.52	0.76
	medial ventral (reuniens)	0.07	0.71	0.91
	anteroventral	0.05	0.79	0.91
	ventral anterior	-0.04	0.81	0.92
	central medial	0.04	0.83	0.93
	whole thalamus	-0.04	0.86	0.93
	centromedian	0.03	0.9	0.94
	parafascicular	-0.02	0.92	0.94
hippocampus	hippocampal fissure	0.48	0.024	0.1
	hippocampal amygdala transition area	0.23	0.22	0.51
	fimbria	0.20	0.35	0.62
	CA1	0.17	0.44	0.71
	CA4	0.13	0.5	0.76
	CA2/3	0.10	0.62	0.9
	parasubiculum	0.10	0.67	0.91
	presubiculum	-0.19	0.67	0.91
	subiculum	-0.13	0.68	0.91
	GC ML DG	0.06	0.77	0.91
	whole hippocampus	0.08	0.79	0.91
	molecular layer	0.01	0.97	0.98
	hippocampal tail	0.00	0.99	0.99
amygdala	cortical nucleus	-0.37	0.048	0.17
	medial nucleus	-0.37	0.08	0.27
	anterior amygdaloid area	-0.17	0.31	0.59
	corticoamygdaloid transition	0.07	0.7	0.91
	basal nucleus	0.10	0.71	0.91
	paralaminar nucleus	0.79	0.75	0.91
	central nucleus	-0.07	0.75	0.91
	accessory basal nucleus	0.06	0.77	0.91
	whole amygdala	0.04	0.88	0.94
	lateral nucleus	-0.02	0.9	0.94

```
# gtsave(group_compare_dup_out, filename = file.path(project,
 ↪  "figures/gene_dosage/dup_vs_td_volumes.png"))
# gtsave(group_compare_dup_out, filename = file.path(project,
 ↪  "figures/gene_dosage/dup_vs_td_volumes.pdf"))
# gtsave(group_compare_dup_out, filename = file.path(project,
 ↪  "figures/gene_dosage/dup_vs_td_volumes.rtf"))
```

age group difference table

```
# export table full age range
age_gdiff_out <- maturation_full$age_group_export_final %>% gt() %>%
  tab_style(style = cell_text(weight = "bold"), locations =
   ↪  list(cells_column_labels())) %>%
  cols_align(align= "right", columns =everything()) %>%
  tab_options(data_row.padding = px(1))
age_gdiff_out
```

Structure	Region	diff_TD_22qDel	diff_TD_22qDup
whole brain	total ICV		
thalamus	anteroventral		
	laterodorsal		
	medial geniculate		
	medial ventral (reuniens)		
	mediodorsal		
	parafascicular	12.9-23.2|34.3-49.5	
	ventral anterior		
	ventromedial	14.2-20.3|49.3-49.5	
hippocampus	CA1		
	CA2/3	5.5-8.8|14.7-26.1	5.5-10.3|14-27.5|43.1-49.5
	CA4		
	GC ML DG	17.5-20.3	
	hippocampal amygdala transition area		
	hippocampal tail		
	molecular layer	18.5-20.1	
	subiculum		
	whole hippocampus		
amygdala	accessory basal nucleus	13.4-22.5|37.1-49.5	
	basal nucleus	14.1-24|45.2-49.5	
	central nucleus	11.1-18.5|28.8-49.5	
	corticoamygdaloid transition	17.5-21.6	
	lateral nucleus	5.5-9.8|14.5-26.9	15.1-22.6
	whole amygdala	5.5-6.6|13.7-24.7|39.4-49.5	

```
# gtsave(age_gdiff_out, filename = file.path(project,
 ↪  "figures/age/age_gdiff_full.png"))
# gtsave(age_gdiff_out, filename = file.path(project,
 ↪  "figures/age/age_gdiff_full.pdf"))
# gtsave(age_gdiff_out, filename = file.path(project,
 ↪  "figures/age/age_gdiff_full.rtf"))

# export table <35
age_gdiff_u35_out <- maturation_young$age_group_export_final %>% gt() %>%
```

```
  tab_style(style = cell_text(weight = "bold"), locations =
   ↪  list(cells_column_labels())) %>%
  cols_align(align= "right", columns =everything()) %>%
  tab_options(data_row.padding = px(1))
age_gdiff_u35_out
```

Structure	Region	diff_TD_22qDel	diff_TD_22qDup
whole brain	total ICV		
thalamus	anteroventral	5.5-8.5|13.4-20.6	
	laterodorsal		
	medial geniculate		
	parafascicular	5.5-8.7|18.8-20.7	
	ventromedial		
hippocampus	CA1		
	CA2/3	5.5-8.8|16-20.9	5.5-9.4|13.6-21.3
	CA4		
	GC ML DG		
	hippocampal fissure		
	hippocampal tail		
	molecular layer		
	presubiculum		
	subiculum		
	whole hippocampus		
amygdala	accessory basal nucleus		
	anterior amygdaloid area		
	basal nucleus		
	central nucleus	11.4-17.5|24.1-34.4	
	corticoamygdaloid transition		
	lateral nucleus		
	paralaminar nucleus		5.5-6.1|14.8-19.7
	whole amygdala		

```
# gtsave(age_gdiff_u35_out, filename = file.path(project,
 ↪  "figures/age/age_gdiff_u35.png"))
# gtsave(age_gdiff_u35_out, filename = file.path(project,
 ↪  "figures/age/age_gdiff_u35.pdf"))
# gtsave(age_gdiff_u35_out, filename = file.path(project,
 ↪  "figures/age/age_gdiff_u35.rtf"))
```

sex interactions

```
# test random intercept model at each region
gam_combat_sexint <- lapply(sc_names_normed, function(r) gam(formula =
 ↪  reformulate( c("s(AGE, by =SUBJECT_IDENTITY, bs =\"tp\", k= 3, fx =TRUE)",
 ↪  "gene_dosage*SEX", "eTIVnormed", "site", "s(SUBJECTID, bs =\"re\", k= 3)"),
 ↪  response = r), data =demo_combat_na_gam, REML=TRUE))
names(gam_combat_sexint) <- sc_names_normed

# get p-val for gene dosage effect
sexint <- lapply(gam_combat_sexint, function(l)
 ↪  summary(l)$p.table["gene_dosage:SEX", c("Estimate", "Pr(>|t|)")]) %>%
 ↪  do.call(rbind,.) %>% as.data.frame
```

```
colnames(sexint) <- c("sexint_beta", "sexint_p")
sexint$Region <- sc_names_normed

# FDR correct
sexint$sexint_fdr_q <- p.adjust(sexint$sexint_p, method = "fdr")
sexint$fdr_sig <- sexint$sexint_fdr_q < 0.05
```

sex main effects

```
# get p-val for gene dosage effect
sex_main <- lapply(gamm_combat, function(l) summary(l)$p.table["SEX",
 ↪  c("Estimate", "Pr(>|t|)")]) %>% do.call(rbind,.) %>% as.data.frame
colnames(sex_main) <- c("sex_beta", "sex_p")
sex_main$Region <- all_names_normed

# FDR correct
sex_main$sex_fdr_q <- p.adjust(sex_main$sex_p, method = "fdr")
sex_main$fdr_sig <- sex_main$sex_fdr_q < 0.05

# create data frame for export
save_sex_effect <- sex_main[, c("Region", "sex_beta", "sex_p", "sex_fdr_q",
 ↪  "fdr_sig", "Region")]

# get region names for matching
save_sex_effect$Region <- gsub(".combat.norm", "", save_sex_effect$Region)
save_sex_effect$Region <- gsub("Thal_", "", save_sex_effect$Region)
save_sex_effect$Region <- gsub("Amy_", "", save_sex_effect$Region)
save_sex_effect$Region <- gsub("Hip_", "", save_sex_effect$Region)

# merge with full names
save_sex_effect <- merge(x =save_sex_effect, y =lut_unique[, c("Structure",
 ↪  "bilat_name", "bilat_match")], by.x = "Region", by.y = "bilat_match", all.x
 ↪  =TRUE)

# set order
save_sex_effect$struct_order <- save_sex_effect$Structure %>% gsub("whole
 ↪  brain",1,.) %>% gsub("thalamus",2,.) %>% gsub("hippocampus",3,.) %>%
 ↪  gsub("amygdala",4,.)

save_sex_effect <- save_sex_effect[with(save_sex_effect, order(struct_order,
 ↪  sex_beta)),]

save_sex_effect$Region <- save_sex_effect$bilat_name
save_sex_effect$Region <- save_sex_effect$Region %>% gsub("hippocampal amygdala
 ↪  transition area", "HATA", .)
rownames(save_sex_effect) <- NULL

# get only significant
save_sex_effect_final <- filter(save_sex_effect[, c("Structure", "Region",
 ↪  "sex_beta", "sex_p", "sex_fdr_q")], sex_fdr_q < 0.05)
colnames(save_sex_effect_final) <-c("Structure", "Region", "beta", "p", "FDR q")

# edit structure names
structure_dup <- duplicated(save_sex_effect_final$Structure)
```

```
for(i in 1:length(structure_dup)){
  if(structure_dup[i]==TRUE){
    save_sex_effect_final[i,"Structure"] <- ""
  }
}

# round
save_sex_effect_final$beta %<>% round(., digits = 2) %>% sprintf("%.2f",.)
save_sex_effect_final$p %<>% formatC(., format = "e", digits = 1)
save_sex_effect_final$`FDR q` %<>% formatC(., format = "g", digits = 2)

# export table
save_sex_effect_out <-save_sex_effect_final%>% gt() %>%
  tab_style(style = cell_text(weight = "bold"), locations =
   ↪  list(cells_column_labels())) %>%
  cols_align(align= "right", columns =everything()) %>%
  tab_options(data_row.padding = px(1))
save_sex_effect_out
```

Structure	Region	beta	p	FDR q
whole brain	total ICV	1.09	7.0e-17	2.9e-15
thalamus	ventromedial	0.32	1.2e-02	0.021
	mediodorsal	0.32	1.4e-02	0.025
	anteroventral	0.33	2.6e-02	0.04
	ventral anterior	0.34	3.6e-03	0.0076
	parafascicular	0.36	1.9e-03	0.0045
	ventral lateral	0.38	1.2e-03	0.0032
	centromedian	0.41	1.1e-03	0.0032
	whole thalamus	0.54	2.2e-05	0.00031
	pulvinar	0.60	1.2e-04	0.00073
hippocampus	CA2/3	0.31	2.5e-02	0.04
	subiculum	0.37	5.1e-03	0.01
	parasubiculum	0.38	1.8e-02	0.031
	fimbria	0.44	3.0e-03	0.0067
	molecular layer	0.47	1.2e-03	0.0032
	CA4	0.48	6.4e-04	0.0022
	CA1	0.49	1.5e-03	0.0037
	HATA	0.50	6.4e-04	0.0022
	GC ML DG	0.50	4.0e-04	0.0017
	whole hippocampus	0.52	1.9e-04	0.0009
	presubiculum	0.57	7.4e-05	0.00052
amygdala	accessory basal nucleus	0.39	5.3e-03	0.01
	corticoamygdaloid transition	0.46	7.6e-04	0.0024
	basal nucleus	0.50	1.7e-04	0.0009
	whole amygdala	0.51	6.9e-05	0.00052
	lateral nucleus	0.54	4.3e-05	0.00045
	paralaminar nucleus	0.64	4.3e-06	9e-05

```
# gtsave(save_sex_effect_out, filename = file.path(project,
 ↪  "figures/gene_dosage/sex_main_effect.png"))
# gtsave(save_sex_effect_out, filename = file.path(project,
 ↪  "figures/gene_dosage/sex_main_effect.pdf"))
```

```
# gtsave(save_sex_effect_out, filename = file.path(project,
 ↪  "figures/gene_dosage/sex_main_effect.rtf"))
```

antipsychotic drug analysis

```
# get t/f column for antipsychotic drug
df_demo$APD <- df_demo$psych_meds == "antipsychotic"

demo_normed_apd <- merge(x =demo_combat_na_gam, y =df_demo[, c("MRI_S_ID",
 ↪  "APD")], by = "MRI_S_ID")

# test random intercept model including APD at each region
gam_combat_apd <- lapply(sc_names_normed, function(r) gam(formula = reformulate(
 ↪  c("s(AGE, by =SUBJECT_IDENTITY, bs =\"tp\", k= 3, fx =TRUE)", "gene_dosage",
 ↪  "APD", "SEX", "eTIVnormed", "site", "s(SUBJECTID, bs =\"re\", k= 3)"),
 ↪  response = r), data =demo_normed_apd, REML=TRUE))
names(gam_combat_apd) <- sc_names_normed

# get p-val for gene dosage effect
gene_dosage_effect_ap <- lapply(gam_combat_apd, function(l)
 ↪  summary(l)$p.table["gene_dosage", c("Estimate", "Pr(>|t|)")]) %>%
 ↪  do.call(rbind,.) %>% as.data.frame
colnames(gene_dosage_effect_ap) <- c("gene_dosage_beta", "gene_dosage_p")
gene_dosage_effect_ap$Region <- sc_names_normed

# FDR correct
gene_dosage_effect_ap$gene_dosage_fdr_q <-
 ↪  p.adjust(gene_dosage_effect_ap$gene_dosage_p, method = "fdr")
gene_dosage_effect_ap$fdr_sig <- gene_dosage_effect_ap$gene_dosage_fdr_q < 0.05

# get only significant effects
gene_dosage_effect_sig_ap <- filter(gene_dosage_effect_ap, fdr_sig==TRUE)
gene_dosage_effect_sig_ap <-
 ↪  gene_dosage_effect_sig_ap[order(gene_dosage_effect_sig_ap$gene_dosage_beta),]

# create data frame for export
save_dosage_effect_ap <- gene_dosage_effect_sig_ap[, c("gene_dosage_beta",
 ↪  "gene_dosage_p", "gene_dosage_fdr_q", "Region")]
# add whole amydgala and thalamus despite no FDR significance
#save_dosage_effect <- rbind(save_dosage_effect,
 ↪  gene_dosage_effect[c("Thal_Whole_thalamus.combat.norm",
 ↪  "Amy_Whole_amygdala.combat.norm"), c("gene_dosage_beta", "gene_dosage_p",
 ↪  "gene_dosage_fdr_q", "Region")])
#colnames(save_dosage_effect) <- c("beta", "p", "FDR q", "Region")

# get region names for matching
save_dosage_effect_ap$Region <- gsub(".combat.norm", "",
 ↪  save_dosage_effect_ap$Region)
save_dosage_effect_ap$Region <- gsub("Thal_", "", save_dosage_effect_ap$Region)
save_dosage_effect_ap$Region <- gsub("Amy_", "", save_dosage_effect_ap$Region)
save_dosage_effect_ap$Region <- gsub("Hip_", "", save_dosage_effect_ap$Region)

# merge with full names
#save_dosage_effect <- merge(x =save_dosage_effect, y =lut[c("Structure", "Name",
 ↪  "region_match")], by.x = "Region", by.y = "region_match", all.x =TRUE)
```

```
save_dosage_effect_ap <- merge(x =save_dosage_effect_ap, y =lut_unique[,
 ↪  c("Structure", "bilat_name", "bilat_match")], by.x = "Region", by.y =
 ↪  "bilat_match", all.x =TRUE)

# set order
save_dosage_effect_ap$struct_order <- save_dosage_effect_ap$Structure %>%
 ↪  gsub("thalamus",1,.) %>% gsub("hippocampus",2,.) %>% gsub("amygdala",3,.)

save_dosage_effect_ap <- save_dosage_effect_ap[with(save_dosage_effect_ap,
 ↪  order(struct_order, gene_dosage_beta)),]

#save_dosage_effect$Structure <- gsub("thalamus", "thal",
 ↪  save_dosage_effect$Structure)
#save_dosage_effect$Structure <- gsub("hippocampus", "hip",
 ↪  save_dosage_effect$Structure)
#save_dosage_effect$Structure <- gsub("amygdala", "amy",
 ↪  save_dosage_effect$Structure)
#save_dosage_effect$Region <- save_dosage_effect$Name
save_dosage_effect_ap$Region <- save_dosage_effect_ap$bilat_name
save_dosage_effect_ap$beta <- save_dosage_effect_ap$gene_dosage_beta
save_dosage_effect_ap$p <- save_dosage_effect_ap$gene_dosage_p
save_dosage_effect_ap$`FDR q` <- save_dosage_effect_ap$gene_dosage_fdr_q
rownames(save_dosage_effect_ap) <- NULL

# round
save_dosage_effect_ap$beta %<>% round(., digits = 2) %>% sprintf("%.2f",.)
#save_dosage_effect$p %<>% round(., digits = 3) %>% sprintf("%.3f",.)
save_dosage_effect_ap$p %<>% formatC(., format = "e", digits = 1)
#save_dosage_effect$`FDR q` %<>% round(., digits = 6) %>% sprintf("%.6f",.)
# g option formats as scientific only when saves space
save_dosage_effect_ap$`FDR q` %<>% formatC(., format = "g", digits = 2)

# edit structure names
structure_dup <- duplicated(save_dosage_effect_ap$Structure)
for(i in 1:length(structure_dup)){
  if(structure_dup[i]==TRUE){
    save_dosage_effect_ap[i,"Structure"] <- ""
  }
}

save_dosage_effect_ap_final <- save_dosage_effect_ap[, c("Structure", "Region",
 ↪  "beta", "p", "FDR q")]
#write.csv(save_dosage_effect_ap_final, file =file.path(project,
 ↪  "figures/gene_dosage/gene_dosage_subregions_apd.csv"), row.names = FALSE)

# export table
save_dosage_effect_ap_out <-save_dosage_effect_ap_final%>% gt() %>%
  tab_style(style = cell_text(weight = "bold"), locations =
   ↪  list(cells_column_labels())) %>%
  cols_align(align= "right", columns =everything()) %>%
  tab_options(data_row.padding = px(1))
save_dosage_effect_ap_out
```

Structure	Region	beta	p	FDR q
thalamus	mediodorsal	-0.35	2.6e-05	0.00036
	lateral geniculate	0.37	1.1e-04	0.00076
	medial ventral (reuniens)	0.40	9.1e-05	0.00075
hippocampus	CA4	0.43	1.2e-03	0.0046
	GC ML DG	0.43	1.0e-03	0.0042
	whole hippocampus	0.49	2.9e-04	0.0013
	CA1	0.50	2.1e-04	0.0012
	subiculum	0.52	4.6e-05	0.00048
	molecular layer	0.52	2.8e-04	0.0013
	hippocampal fissure	0.56	5.1e-08	2.1e-06
	hippocampal tail	0.62	3.6e-07	7.3e-06
amygdala	paralaminar nucleus	0.31	3.2e-03	0.01
	basal nucleus	0.34	2.6e-03	0.0089

```
# gtsave(save_dosage_effect_ap_out, filename = file.path(project,
 ↪  "figures/gene_dosage/gene_dosage_antipsychotic.png"))
# gtsave(save_dosage_effect_ap_out, filename = file.path(project,
 ↪  "figures/gene_dosage/gene_dosage_antipsychotic.pdf"))
# gtsave(save_dosage_effect_ap_out, filename = file.path(project,
 ↪  "figures/gene_dosage/gene_dosage_antipsychotic.rtf"))

# test random intercept model of APD in only 22qDel
ap_model <- lapply(sc_names_normed, function(r) gam(formula = reformulate(c("APD",
 ↪  "s(AGE, by =SUBJECT_IDENTITY, bs =\"tp\", k= 3, fx =TRUE)", "SEX",
 ↪  "eTIVnormed", "site", "s(SUBJECTID, bs =\"re\", k= 3)"), response = r), data
 ↪  =filter(demo_normed_apd, SUBJECT_IDENTITY== "PATIENT-DEL"), REML=TRUE))
names(ap_model) <- sc_names_normed
ap_effect <- lapply(ap_model, function(l) summary(l)$p.table["APDTRUE",
 ↪  c("Estimate", "Pr(>|t|)")]) %>% do.call(rbind,.) %>% as.data.frame
colnames(ap_effect) <- c("beta", "p")
ap_effect$Region <- sc_names_normed
ap_effect$fdr_q <- p.adjust(ap_effect$p, method = "fdr")
ap_effect$fdr_sig <- ap_effect$fdr_q < 0.05
# no regions with FDR q < 0.05
```

verbal IQ comparison with hippocampal tail (Latreche et al 2023) https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10476015/

```
demo_sx_mri <- merge(x =demo_combat_na_gam, y =df_demo[, c("MRI_S_ID",
 ↪  "WASI_verbal", "WASI_matrix", "IQ_full")], by = "MRI_S_ID")

# function to test formula on data
test_sx_lmm <- function(formula, data){
  mod <- lmerTest::lmer(formula, data =data, REML=TRUE, na.action= "na.omit")
  modsum <- summary(mod)
  out <- modsum$coefficients
  return(out)
}

# verbal IQ vs hipp tail in each group
print("Verbal IQ:")
```

```
## [1] "Verbal IQ:"
```

```
groups <- c("PATIENT-DEL", "PATIENT-DUP", "CONTROL")
hipp_tail_viq <- lapply(groups, function(g) test_sx_lmm(formula = "WASI_verbal ~
 ↪  Hip_Hippocampal_tail.combat.norm + SEX + site  + (1|SUBJECTID)", data
 ↪  =filter(demo_sx_mri, SUBJECT_IDENTITY==g)))
names(hipp_tail_viq) <- groups
hipp_tail_viq
```

```
## $`PATIENT-DEL`
##                                  Estimate Std. Error        df    t value
## (Intercept)                     37.691814  1.5817466 109.19597 23.8292375
## Hip_Hippocampal_tail.combat.norm  1.860354  0.7212128 130.96120  2.5794800
## SEX                               1.030443  1.8006149  89.61745  0.5722727
## site                              4.373378  1.0133889 147.85197  4.3155969
##                                     Pr(>|t|)
## (Intercept)                     4.527159e-45
## Hip_Hippocampal_tail.combat.norm 1.099840e-02
## SEX                              5.685701e-01
## site                             2.899232e-05
##
## $`PATIENT-DUP`
##                                   Estimate Std. Error       df    t value
## (Intercept)                     47.6449092   3.209040 39.35927 14.8470898
## Hip_Hippocampal_tail.combat.norm  2.5438232   1.688942 57.57944  1.5061640
## SEX                              -2.4372811   4.234338 34.31285 -0.5755991
## site                             -0.8577922   1.391474 31.98451 -0.6164631
##                                     Pr(>|t|)
## (Intercept)                     1.014745e-17
## Hip_Hippocampal_tail.combat.norm 1.374919e-01
## SEX                              5.686422e-01
## site                             5.419521e-01
##
## $CONTROL
##                                   Estimate Std. Error        df    t value
## (Intercept)                     62.7656754   2.144863  86.47614 29.2632603
## Hip_Hippocampal_tail.combat.norm -0.9366697   1.310072 101.07921 -0.7149759
## SEX                              -6.0389277   2.791619  74.01856 -2.1632346
## site                             -3.9322620   2.286445 118.47454 -1.7198151
##                                     Pr(>|t|)
## (Intercept)                     1.235490e-46
## Hip_Hippocampal_tail.combat.norm 4.762721e-01
## SEX                              3.375205e-02
## site                             8.807729e-02
```

```
# matrix IQ vs hipp tail in each group
print("Matrix IQ:")
```

```
## [1] "Matrix IQ:"
```

```
groups <- c("PATIENT-DEL", "PATIENT-DUP", "CONTROL")
hipp_tail_miq <- lapply(groups, function(g) test_sx_lmm(formula = "WASI_matrix ~
 ↪  Hip_Hippocampal_tail.combat.norm + SEX + site  + (1|SUBJECTID)", data
 ↪  =filter(demo_sx_mri, SUBJECT_IDENTITY==g)))
names(hipp_tail_miq) <- groups
```

```
hipp_tail_miq
```

```
## $`PATIENT-DEL`
##                                   Estimate Std. Error        df    t value
## (Intercept)                     33.2186204   1.921771 111.67110 17.2854187
## Hip_Hippocampal_tail.combat.norm -0.5652688   0.875984 131.59252 -0.6452958
## SEX                               1.9855327   2.168126  92.60277  0.9157828
## site                              3.7276525   1.254189 151.59184  2.9721614
##                                      Pr(>|t|)
## (Intercept)                     2.394472e-33
## Hip_Hippocampal_tail.combat.norm 5.198591e-01
## SEX                              3.621595e-01
## site                             3.440448e-03
##
## $`PATIENT-DUP`
##                                   Estimate Std. Error       df    t value
## (Intercept)                     47.2607877   3.173685 46.41537 14.8914545
## Hip_Hippocampal_tail.combat.norm  1.5421672   1.791183 43.76239  0.8609770
## SEX                               0.5024728   3.846671 33.72805  0.1306254
## site                              1.1643918   2.172564 40.14848  0.5359527
##                                      Pr(>|t|)
## (Intercept)                     2.674445e-19
## Hip_Hippocampal_tail.combat.norm 3.939435e-01
## SEX                              8.968477e-01
## site                             5.949483e-01
##
## $CONTROL
##                                   Estimate Std. Error        df    t value
## (Intercept)                     55.8543673   1.893697  85.22077 29.4948779
## Hip_Hippocampal_tail.combat.norm -0.2496773   1.172768  91.90799 -0.2128957
## SEX                              -3.4123523   2.424433  74.16283 -1.4074849
## site                              0.5301897   2.177880 121.91154  0.2434431
##                                      Pr(>|t|)
## (Intercept)                     1.714465e-46
## Hip_Hippocampal_tail.combat.norm 8.318798e-01
## SEX                              1.634612e-01
## site                             8.080714e-01
```

```
# verbal controlling for matrix IQ vs hipp tail in each group
print("Verbal controlling for Matrix IQ:")
```

```
## [1] "Verbal controlling for Matrix IQ:"
```

```
groups <- c("PATIENT-DEL", "PATIENT-DUP", "CONTROL")
hipp_tail_ciq <- lapply(groups, function(g) test_sx_lmm(formula = "WASI_verbal ~
 ↪  Hip_Hippocampal_tail.combat.norm + WASI_matrix + SEX + site  + (1|SUBJECTID)",
 ↪  data =filter(demo_sx_mri, SUBJECT_IDENTITY==g)))
names(hipp_tail_ciq) <- groups
hipp_tail_ciq
```

```
## $`PATIENT-DEL`
##                                   Estimate Std. Error        df    t value
## (Intercept)                     28.9652286 2.38231143 155.37375 12.1584560
## Hip_Hippocampal_tail.combat.norm  2.0722725 0.64696712 109.73894  3.2030570
```

```
## WASI_matrix                          0.2670440 0.05761027 177.65798  4.6353542
## SEX                                  0.5434794 1.56631125  80.63078  0.3469804
## site                                 3.2360671 1.03395651 159.86165  3.1297903
##                                         Pr(>|t|)
## (Intercept)                          2.523368e-24
## Hip_Hippocampal_tail.combat.norm     1.779938e-03
## WASI_matrix                          6.878185e-06
## SEX                                  7.295100e-01
## site                                 2.079898e-03
##
## $`PATIENT-DUP`
##                                        Estimate Std. Error       df    t value
## (Intercept)                          36.3005557  5.5194857 56.34834  6.5768005
## Hip_Hippocampal_tail.combat.norm      2.4827804  1.5953844 48.26318  1.5562271
## WASI_matrix                           0.2441939  0.1005468 47.04098  2.4286578
## SEX                                  -2.7225259  3.7187127 25.93705 -0.7321152
## site                                 -1.3579982  1.5187235 26.04776 -0.8941708
##                                         Pr(>|t|)
## (Intercept)                          1.684495e-08
## Hip_Hippocampal_tail.combat.norm     1.261879e-01
## WASI_matrix                          1.903074e-02
## SEX                                  4.706625e-01
## site                                 3.794186e-01
##
## $CONTROL
##                                        Estimate Std. Error        df    t value
## (Intercept)                          26.8325617 4.68275555 115.83591  5.7300795
## Hip_Hippocampal_tail.combat.norm     -0.4473155 0.99523829  82.45147 -0.4494557
## WASI_matrix                           0.6419459 0.07890426 119.61362  8.1357582
## SEX                                  -3.8389362 2.05347527  68.00817 -1.8694826
## site                                 -3.8275232 1.89718523 119.37816 -2.0174747
##                                         Pr(>|t|)
## (Intercept)                          8.086510e-08
## Hip_Hippocampal_tail.combat.norm     6.542823e-01
## WASI_matrix                          4.370185e-13
## SEX                                  6.586147e-02
## site                                 4.588874e-02
```

plot verbal IQ vs hippocampus tail

```
dat <- demo_sx_mri
# function to plot lmm on top of data
scatterplot_lmm <- function(group, x, y, title = "", xlab= "", ylab= "", linecol =
↪  "black", pval = "missing"){
  dat <- filter(demo_sx_mri, SUBJECT_IDENTITY==group)
  # get predicted values for model
  mod <- effect(term= x, mod =lme4::lmer(formula = as.formula(paste(y,"~", x,"+
↪  SEX + site + (1|SUBJECTID)")), data =dat, REML=TRUE)) %>% as.data.frame
  # plot
  ggplot(data =dat, aes_string(x = x, y = y))+
    geom_point()+
    geom_ribbon(data =mod, inherit.aes = FALSE, aes_string(x = x, ymin= "lower",
     ↪  ymax = "upper"), fill = "grey", alpha = 0.6)+
    geom_line(data =mod, inherit.aes = FALSE, aes_string(x = x, y = "fit",
     ↪  group=NA), color =linecol)+
```

```
    ggtitle(title)+
    xlab(xlab)+
    ylim(10,85)+
    xlim(300,800)+
    #scale_y_continuous(breaks =seq(20,80, by = 20))+
    ylab(ylab)+
    theme_classic()+
    annotate(geom = "text", x = 400, y = 83, label = paste0("p= ", pval), size =
     ↪  4, color = "black")
}

del_viq_plot <- scatterplot_lmm(group= "PATIENT-DEL", x =
 ↪  "Hip_Hippocampal_tail.combat", y = "WASI_verbal", title = "22qDel", xlab=
 ↪  "Hippocampal tail volume", ylab= "Verbal IQ", linecol = "blue", pval =
 ↪  round(hipp_tail_viq$`PATIENT-DEL`["Hip_Hippocampal_tail.combat.norm",
 ↪  "Pr(>|t|)"], digits = 3))

dup_viq_plot <- scatterplot_lmm(group= "PATIENT-DUP", x =
 ↪  "Hip_Hippocampal_tail.combat", y = "WASI_verbal", title = "22qDup", pval =
 ↪  round(hipp_tail_viq$`PATIENT-DUP`["Hip_Hippocampal_tail.combat.norm",
 ↪  "Pr(>|t|)"], digits = 3))

td_viq_plot <- scatterplot_lmm(group= "CONTROL", x =
 ↪  "Hip_Hippocampal_tail.combat", y = "WASI_verbal", title = "Control", pval =
 ↪  round(hipp_tail_viq$`CONTROL`["Hip_Hippocampal_tail.combat.norm", "Pr(>|t|)"],
 ↪  digits = 3))

all_viq_plot <- del_viq_plot + dup_viq_plot + td_viq_plot +
 ↪  plot_annotation(tag_levels = 'A')& theme(plot.tag = element_text(face =
 ↪  'bold'))

#ggsave(plot= all_viq_plot, filename =file.path(project,
 ↪  "figures/cognition/hip_tail_verbal_iq.pdf"), width= 9, height= 5, device =
 ↪  "pdf")
#ggsave(plot= all_viq_plot, filename =file.path(project,
 ↪  "figures/cognition/hip_tail_verbal_iq.png"), width= 9, height= 5, device =
 ↪  "png", dpi = 300)
```

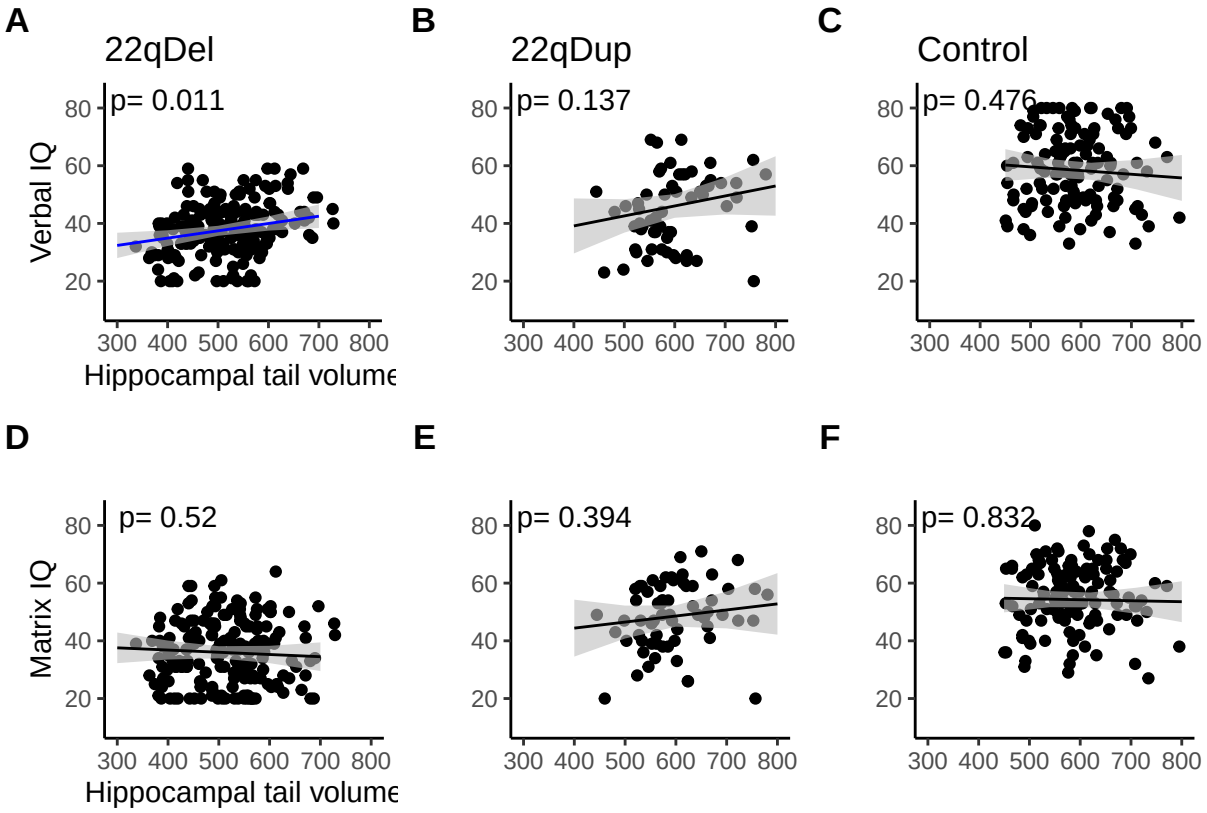
same for matrix IQ

```
del_miq_plot <- scatterplot_lmm(group= "PATIENT-DEL", x =
 ↪  "Hip_Hippocampal_tail.combat", y = "WASI_matrix", title = "", xlab=
 ↪  "Hippocampal tail volume", ylab= "Matrix IQ", pval =
 ↪  round(hipp_tail_miq$`PATIENT-DEL`["Hip_Hippocampal_tail.combat.norm",
 ↪  "Pr(>|t|)"], digits = 3))

dup_miq_plot <- scatterplot_lmm(group= "PATIENT-DUP", x =
 ↪  "Hip_Hippocampal_tail.combat", y = "WASI_matrix", title = "", pval =
 ↪  round(hipp_tail_miq$`PATIENT-DUP`["Hip_Hippocampal_tail.combat.norm",
 ↪  "Pr(>|t|)"], digits = 3))

td_miq_plot <- scatterplot_lmm(group= "CONTROL", x =
 ↪  "Hip_Hippocampal_tail.combat", y = "WASI_matrix", title = "", pval =
 ↪  round(hipp_tail_miq$`CONTROL`["Hip_Hippocampal_tail.combat.norm", "Pr(>|t|)"],
 ↪  digits = 3))
```

```
all_miq_plot <- del_miq_plot + dup_miq_plot + td_miq_plot +
 ↪  plot_annotation(tag_levels = 'A')& theme(plot.tag = element_text(face =
 ↪  'bold'))
#all_miq_plot
```

plot both together

```
all_iq_plot <- all_viq_plot / all_miq_plot + plot_annotation(tag_levels = 'A')&
 ↪  theme(plot.tag = element_text(face = 'bold'))
all_iq_plot
```




```
#ggsave(plot= all_iq_plot, filename =file.path(project,
 ↪  "figures/cognition/hip_tail_iq.pdf"), width= 9, height= 7, device = "pdf")
#ggsave(plot= all_iq_plot, filename =file.path(project,
 ↪  "figures/cognition/hip_tail_iq.png"), width= 9, height= 7, device = "png", dpi
 ↪  = 300)
```

test model of SIPS scores vs volume in 22qDel

```
# test random intercept model of SIPS positive score in only 22qDel
demo_normed_sips <- merge(x =demo_combat_normed, y =df_demo[, c("MRI_S_ID",
 ↪  "SIPS_p_sum", "SIPS_prodromal")], by = "MRI_S_ID")
#sipsp_model <- lapply(sc_names_normed, function(r) lmerTest::lmer(formula =
 ↪  reformulate(c("SIPS_p_sum", "AGE", "AGE2", "SEX", "eTIVnormed", "site",
 ↪  "(1|SUBJECTID)"), response = r), data =filter(demo_normed_sips,
 ↪  SUBJECT_IDENTITY== "PATIENT-DEL"), REML=TRUE))
sipsp_model <- lapply(sc_names_normed, function(r) lmerTest::lmer(formula =
 ↪  as.formula(paste("SIPS_p_sum ~ AGE + AGE2 + SEX + eTIVnormed + site +
 ↪  (1|SUBJECTID) +", r)), data =filter(demo_normed_sips, SUBJECT_IDENTITY==
 ↪  "PATIENT-DEL"), REML=TRUE))
```

```
names(sipsp_model) <- sc_names_normed
#sipsp_effect <- lapply(sipsp_model, function(l) summary(l, ddf=
 ↪  "Kenward-Roger")$coefficients["SIPS_p_sum", c("Estimate", "Pr(>|t|)")]) %>%
 ↪  do.call(rbind,.) %>% as.data.frame
sipsp_effect <- lapply(names(sipsp_model), function(l) summary(sipsp_model[[l]],
 ↪  ddf= "Kenward-Roger")$coefficients[l, c("Estimate", "Pr(>|t|)")]) %>%
 ↪  do.call(rbind,.) %>% as.data.frame
colnames(sipsp_effect) <- c("beta", "p")
sipsp_effect$Region <- sc_names_normed
sipsp_effect$fdr_q <- p.adjust(sipsp_effect$p, method = "fdr")
sipsp_effect$fdr_sig <- sipsp_effect$fdr_q < 0.05
# no significant results

# test random intercept model of SIPS prodromal status
sipspr_model <- lapply(sc_names_normed, function(r) lmerTest::lmer(formula =
 ↪  reformulate(c("SIPS_prodromal", "AGE", "AGE2", "SEX", "eTIVnormed", "site",
 ↪  "(1|SUBJECTID)"), response = r), data =filter(demo_normed_sips,
 ↪  SUBJECT_IDENTITY== "PATIENT-DEL"), REML=TRUE))
names(sipspr_model) <- sc_names_normed
sipspr_effect <- lapply(sipspr_model, function(l) summary(l, ddf=
 ↪  "Kenward-Roger")$coefficients["SIPS_prodromalTRUE", c("Estimate",
 ↪  "Pr(>|t|)")]) %>% do.call(rbind,.) %>% as.data.frame
colnames(sipspr_effect) <- c("beta", "p")
sipspr_effect$Region <- sc_names_normed
sipspr_effect$fdr_q <- p.adjust(sipspr_effect$p, method = "fdr")
sipspr_effect$fdr_sig <- sipspr_effect$fdr_q < 0.05
# no significant results
```

save workspace

save package info as package_versions.txt save data Rdata object (excluding full gamm model to save space)

```
# #save list of packages
# setwd(project)
# sink("package_versions.txt")
# sessionInfo()
# sink()
# # list objects by size
# all_obj <- sort( sapply(ls(), function(x){object.size(get(x))})) %>% as.matrix()
# # list any objects over 30000000 B
# large_obj <- all_obj[which(all_obj[,1]>30000000),] %>% names %>% as.vector
# # clear to use less space
# gam_combat_dup <- NULL
# gam_combat_del <- NULL
# gam_combat_sexint <- NULL
# gam_combat_apd <- NULL
# gamm_combat_young <- NULL
# maturation_young <- NULL
# gamm_combat <- NULL
# maturation_full <- NULL
# mat_young_gdiff <- NULL
# mat_full_gdiff <- NULL
```

```
#
# # save image
# save.image(file =file.path(project, "git_exclude/subcort_volumes_data.Rdata"),
 ↪  compress = "bzip2")
#
 ↪  #load("~/Dropbox/github/22q_subcort_volumes/git_exclude/subcort_volumes_data.Rdata")
```