03_STOPPD_masterDF

Navona

This script combines information in XNAT/file system (which have already been established to be identical in script 02_STOPPD_xnat) and study logs, and randomization (recently unblinded), into a single, master spreadsheet.

Purpose: the output csv (STOPPD_participantList_2018-11-05.csv) is meant to serve as a master reference sheet for all participants that were randomized (irrespective of scan completion).

This script now also adds a column relating to whether or not the subject is ok for MR analysis (i.e. not excluded for later identified neurological condition)

Note: this script does not remove individuals who failed preprocessing, QC, or should be removed from the dataset for any other reason.

```
library('stringi')
library('plyr')
library('tidyr')
library('stringr')
```

```
#import spreadsheets
xnat <- read.csv('../generated_csvs/xnat_clean_2018-01-25.csv', stringsAsFactors = FALSE) #generated by
randomization <- read.csv('../data/clinical/randomization.csv', stringsAsFactors = FALSE) #from Judy (S
log <- read.csv('../data/clinical/master_log.csv', fileEncoding="latin1", na.strings=c("","."," ","NA", "N/

#transform XNAT df from long to wide format
xnat <- xnat[!names(xnat) %in% c('contains_R', 'id_session')] #remove unnecessary variables
xnat <- reshape(xnat, idvar = "MR.ID", timevar = 'session', direction = "wide")
```

```
## Warning in reshapeWide(data, idvar = idvar, timevar = timevar, varying =
## varying, : multiple rows match for session=1: first taken
```

```
## Warning in reshapeWide(data, idvar = idvar, timevar = timevar, varying =
## varying, : multiple rows match for session=2: first taken
```

```
colnames(xnat) <- c('subject_id', 'first_date_xnat', 'second_date_xnat', 'third_date_xnat', 'acute_date_
```

```
#merge xnat with randomization - will get rid of controls, etc
df <- merge(randomization[c('STUDYID', 'BLINDMED')], xnat, all.x=TRUE, by.x='STUDYID', by.y = 'subject_

#rename randomization column
colnames(df)[colnames(df)=="BLINDMED"] <- "randomization"

#combine the 'notes' columns in the log file (easier to read for now)
log$Comments.1 <- paste(log$Specify.reason.if.scan.not.completed.1, log$Comments.1)
log$Comments.2 <- paste(log$Specify.reason.if.scan.not.completed.2, log$Comments.2)
log$Comments.3 <- paste(log$Specify.reason.if.scan.not.completed.3, log$Comments.3)

#make subset of log variables from log we want to merge with randomization info
log <- log[c(
  "STOPPD.clinical.Trial.ID.Imaging.ID",
```

```
  'Sex',
  'Age',
  "Date.of.randomization...Stop.PD",
  "Date.of.consent.to.imaging.study",
  "If.not.enrolled.to.imaging.study..specify.reason.",
  "Study.day.of.acute.phase.MRI",
  "Scan.completed.Y.N",
  "Date.of.MRI..1" ,
  "Study.week",
  "Scan.completed.Y.N.1",
  "Comments.1",
  "Date.of.MRI..2",
  "Study.week.1",
  "Scan.completed",
  "Comments.2",
  "Date.of.MRI..3",
  "Study.week.2",
  "Scan.completed.1",
  "Comments.3")]

#rename the columns of the variables from log we want to merge with randomization info, for clarity
colnames(log) <- c(
  'subject_id',
  'sex',
  'age',
  'randomization_date',
  'imaging_consent_date',
  'imaging_nonconsent_reason',
  'acute_date_log',
  'acute_complete_log',
  'first_date_log',
  'first_timepoint_log',
  'first_complete_log',
  'first_notes',
  'second_date_log',
  'second_timepoint_log',
  'second_complete_log',
  'second_notes',
  'third_date_log',
  'third_timepoint_log',
  'third_complete_log',
  'third_notes')

#merge the df and log data
df <- merge(df, log, all.x=TRUE, by.x = 'STUDYID', by.y='subject_id')

#reorder df columns, for clarity
df <- df[c(
  "STUDYID",
  'sex',
  'age',
  "randomization",
  'randomization_date',
```

```
  'imaging_consent_date',
  'imaging_nonconsent_reason',
  "acute_date_log",
  "acute_complete_log",
  "acute_date_xnat",
  "first_date_log",
  "first_timepoint_log",
  "first_complete_log",
  'first_notes',
  "first_date_xnat",
  "second_date_log",
  "second_timepoint_log",
  "second_complete_log",
  'second_notes',
  "second_date_xnat",
  "third_date_log",
  "third_timepoint_log",
  "third_complete_log",
  'third_notes',
  "third_date_xnat")]

#make sure dates, etc. are characters (not factors) by converting all factors to characters
i <- sapply(df, is.factor)
df[i] <- lapply(df[i], as.character)

#clean up the NA-related values (which exist in the 3 notes columsn, 'first_notes', 'second_notes', 'th
df <- data.frame(lapply(df, function(x) {
      gsub("NA NA", NA, x)
      }))

df <- data.frame(lapply(df, function(x) {
      gsub("NA", NA, x)
      }))

#alter incorrect/unclear values as required
  #acute scan
  df$acute_complete_log <- as.character(df$acute_complete_log)
  df$acute_complete_log[df$acute_complete_log == 'Y'] <- "Yes"
  df$acute_complete_log[df$acute_complete_log == "N" & df$STUDYID == '420043'] <- NA #(replace 'no' wit

  #first scan (replace 'no' with NA, to take care of inconsistent notation)
  df["first_complete_log"] <- lapply(df["first_complete_log"], function(x) {
    gsub("No", NA, x)
    })

  #second scan
    df["second_complete_log"] <- lapply(df["second_complete_log"], function(x) {
    gsub("No", NA, x)
    })

  #third scan
  df$third_timepoint_log[df$third_timepoint_log == "what would be RCT Week 36"] <- "RCT Week 36"
```

```r
#remove 'day' information from 'acute_date_log' and turn into integer
df$acute_date_log <- sub('\\,.*', '', df$acute_date_log) #strip out day info
df$acute_date_log <- as.numeric(substr(df$acute_date_log, 11, 12)) #remove number, make numeric
names(df)[names(df) == 'acute_date_log'] <- 'acute_week_log' #change name of variable for clarity

#separate timepoint source and week information in 'first_timepoint_log' variable
df <- cbind(df, as.data.frame(matrix(str_split_fixed(df$first_timepoint_log, " Week ", 2), ncol = 2, by
df <- subset(df, select = -first_timepoint_log)
colnames(df)[colnames(df)=="V1"] <- "first_timepoint_log"
colnames(df)[colnames(df)=="V2"] <- "first_week_log"

#remove accidental extra space in character
df$second_timepoint_log <- as.character(df$second_timepoint_log)
df$second_timepoint_log[df$second_timepoint_log == 'Off protocol '] <- 'Off protocol'

#separate timepoint source and week information in 'second_timepoint_log' variable
df <- cbind(df, as.data.frame(matrix(str_split_fixed(df$second_timepoint_log, " Week ", 2), ncol = 2, by
df <- subset(df, select = -second_timepoint_log )
colnames(df)[colnames(df)=="V1"] <- "second_timepoint_log"
colnames(df)[colnames(df)=="V2"] <- "second_week_log"

#recode anything containing 'relapse' in 'second_timepoint_log' variable as simply 'relapse'
df$second_timepoint_log <- as.character(df$second_timepoint_log)
df$second_timepoint_log <- ifelse(grepl('Relapse', df$second_timepoint_log), "Relapse", df$second_timepo

#recode anything containing 'Protocol' in 'second_timepoint_log' variable as simply 'off protocol'
df$second_timepoint_log <- ifelse(grepl('Protocol', df$second_timepoint_log), "Off protocol", df$second_

#separate timepoint source and week information in 'third_timepoint_log' variable
df <- cbind(df, as.data.frame(matrix(str_split_fixed(df$third_timepoint_log, " Week ", 2), ncol = 2, by
df <- subset(df, select = -third_timepoint_log )
colnames(df)[colnames(df)=="V1"] <- "third_timepoint_log"
colnames(df)[colnames(df)=="V2"] <- "third_week_log"

#compare dates in df that comes from log vs. XNAT (in new column)
df$first_dateDiff <- round(difftime(df$first_date_log, df$first_date_xnat, units = "days"), 2)
df$second_dateDiff <- round(difftime(df$second_date_log, df$second_date_xnat, units = "days"), 2)
df$third_dateDiff <- round(difftime(df$third_date_log, df$third_date_xnat, units = "days"), 2)

#make sure new variables are characters (not factors), and turn blank values into NA
i <- sapply(df, is.factor)
df[i] <- lapply(df[i], as.character)
df[df == ""] <- NA

#calculate the difference in weeks between scan 2 and scan 1 (i.e., calculate 'second week log' when ab
df$dateDiff_first_second <- round(difftime(df$second_date_log, df$first_date_log, units = "weeks"), 0)
df$dateDiff_first_second <- as.numeric(df$dateDiff_first_second) #turn variables into integers
df$first_week_log <- as.numeric(df$first_week_log) #turn variables into integers
df$second_week_log <- ifelse(is.na(df$second_week_log) & !is.na(df$second_timepoint_log), paste(df$datel

#reorder df columns
df <- df[c(
  "STUDYID",
```

```
  'sex',
  'age',
  "randomization",
  'randomization_date',
  'imaging_consent_date',
  'imaging_nonconsent_reason',
  "acute_week_log",
  "acute_complete_log",
  "first_date_log",
  "first_timepoint_log",
  "first_week_log",
  "first_complete_log",
  'first_notes',
  "second_date_log",
  "second_timepoint_log",
  "second_week_log",
  "second_complete_log",
  'second_notes',
  "third_date_log",
  "third_timepoint_log",
  "third_week_log",
  "third_complete_log",
  'third_notes'
)]

#remove '_log' component of all variable names, for clarity
names(df) = gsub(pattern = "_log", replacement = "", x = names(df))
```

Exclusions from MR analysis and reasons

subject 320032 (PMC): incidental findings more atrophy, should excluded **subject 410012 (CMH)**: another incidental finding, case may have effected longitudinal brain morphomentry

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:plyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
df <- df %>% mutate(MR_exclusion = if_else(STUDYID %in% c("320032", "410012"), "Yes", "No"))
```

```
#make a smaller df of minimally necessary information from participants that completed 2 scans, as requ
write.csv(df, '../generated_csvs/STOPPD_masterDF_2018-11-05.csv', row.names=FALSE)
```

```r
#remove participants that don't have 'Yes' in 'first_complete'
df <- df %>% filter(first_complete == "Yes") #nrow = 88, which is correct

#remove participants that don't have 'Yes' in 'second_complete'
df <-df %>% filter(second_complete == "Yes") #nrow = 74, which is correct

#remove redundant columns
df <- df %>% select(STUDYID, age, sex, randomization, MR_exclusion, first_timepoint, second_timepoint,

df <- df %>% dplyr::rename("offlabel_timepoint" = third_timepoint)
```

```r
#write.csv
write.csv(df, '../generated_csvs/STOPPD_participantList_2018-11-05.csv', row.names=FALSE)
```