## R code to run the AD-NMA

###################

# LIBRARIES

###################

library(tidyverse)

library(readxl)

library(gemtc)

library(eoffice)

require(pcnetmeta)

library(dosresmeta)

# library(metafor)

.pardef <- par() # save default graphical parameters

########################

### first load the data

# Put the name of the Excel file where the data is stored

# myfile <- "AD-NMA-2021-11-08\_DP2.xlsx"

ds <- read\_xlsx(myfile, sheet="STUDIES") %>% filter(!is.na(study))

da <- read\_xlsx(myfile, sheet="ARMS") %>% filter(!is.na(study))

do <- read\_xlsx(myfile, sheet="OUTCOMES", col\_types=c("guess", "guess", "text", rep("guess",44))) %>% filter(!is.na(study))

dt <- read\_xlsx(myfile, sheet="TreatmentCodes") %>% filter(!is.na(treatment)) # to rename treatments

dc <- read\_xlsx(myfile, sheet="OutcomeCodes") %>% filter(!is.na(outcome)) # to select outcomes for analysis

# Check all trial names are on the arms data

xx <- filter(ds, !(study %in% da$study))

xx <- filter(ds, !(study %in% do$study))

# check all arms treatment are in the TreatmentCodes data

do[!(do$treatment %in% dt$treatment), c("study","treatment","outcome")] %>% arrange(study,treatment)

## ensure variables are in lower case

ds$tainf <- tolower(ds$tainf)

ds$agegroup <- tolower(ds$agegroup)

# create variable for low risk in all aspects

ds$nlow <- apply(ds[,paste0("rob",c(1:7))], 1, function(x) sum(grepl("low",x,ignore.case=T)))

# ensure numeric variables in do

for(c in c(7:15,17:46)) do[,c,drop=T] <- as.numeric(do[,c, drop=T])

# Add number or events

## Prepare data for CONINUOUS outcomes

# The objective is to calculate the stats of the changes: C.mean, C.sd, C.se

dn <- filter(do, is.na(cases)) %>%

left\_join(transmute(filter(da, ref %in% 1), study, treatment, ref)) %>%

group\_by(study, subgroup, outcome, time) %>% mutate(rn=mean(ref\*n, na.rm=T)) %>% ungroup() %>%

relocate(rn, .after=n) %>%

# estimate SD of change from differences in change between groups (both in original units and

# Standard error of difference of changes

mutate(

DC.se = ifelse(is.na(DC.se), abs(DC.up-DC.lo)/(2\*qnorm((1+DC.ci)/2)), DC.se),

DP.se = ifelse(is.na(DP.se), abs(DP.up-DP.lo)/(2\*qnorm((1+DP.ci)/2)), DP.se),

# Estimate average variance of changes in each comparison

DC.va = (DC.se^2)/(1/n + 1/rn),

DP.va = (DP.se^2)/(1/n + 1/rn)) %>%

# Estimate SD of changes in each treatment in each study

group\_by(study, outcome, time) %>%

mutate(# Estimate variance so changes in ref group

DC.va=mean(DC.va, na.rm=T),

DP.va=mean(DP.va, na.rm=T),

# Refine variance of changes in the other groups

DC.va=ifelse(!is.na(DC.se), n\*((DC.se^2)-(DC.va/rn)), DC.va),

DP.va=ifelse(!is.na(DP.se), n\*((DP.se^2)-(DP.va/rn)), DP.va)) %>%

ungroup() %>%

# First calculate SE, SD, in each time, first from its own data

mutate(

# At baseline

B.se = ifelse(is.na(B.se), B.sd/sqrt(n), B.se),

B.se = ifelse(is.na(B.se), abs(B.up-B.lo)/(2\*qnorm((1+B.ci)/2)), B.se),

B.sd = ifelse(is.na(B.sd), B.se\*sqrt(n), B.sd),

# Follow-up

F.se = ifelse(is.na(F.se), F.sd/sqrt(n), F.se),

F.se = ifelse(is.na(F.se), abs(F.up-F.lo)/(2\*qnorm((1+F.ci)/2)), F.se),

F.sd = ifelse(is.na(F.sd), F.se\*sqrt(n), F.sd),

# Proportions

P.se = ifelse(is.na(P.se), P.sd/sqrt(n), P.se),

P.se = ifelse(is.na(P.se), abs(P.up-P.lo)/(2\*qnorm((1+P.ci)/2)), P.se),

# Get it from differences in proportional changes

P.se = ifelse(is.na(P.se), sqrt(DP.va/n), P.se),

P.sd = ifelse(is.na(P.sd), P.se\*sqrt(n), P.sd),

# Changes

C.se = ifelse(is.na(C.se), C.sd/sqrt(n), C.se),

C.se = ifelse(is.na(C.se), abs(C.up-C.lo)/(2\*qnorm((1+C.ci)/2)), C.se),

# Get it from differences in changes

C.se = ifelse(is.na(C.se), sqrt(DC.va/n), C.se),

# Get it from the proportional changes in changes (also the mean)

C.se = ifelse(is.na(C.se), B.mean\*P.se/100, C.se),

C.mean = ifelse(is.na(C.mean), B.mean\*P.mean/100, C.mean),

# Get it from Follow-up minus Before assuming 0.35 of before-after correlation (seen in data)

C.se = ifelse(is.na(C.se), sqrt(B.se^2 + F.se^2 - 2\*0.35\*B.se\*F.se), C.se),

C.mean = ifelse(is.na(C.mean), F.mean-B.mean, C.mean),

C.sd = ifelse(is.na(C.sd), C.se\*sqrt(n), C.sd)) %>%

transmute(study, subgroup, treatment, smd\_treat=dt$for\_smd[match(treatment, dt$treatment)], smd\_treat=ifelse(is.na(smd\_treat), treatment, smd\_treat), outcome, time, weeks, rep, n, mean=C.mean, std.err=C.se, sd=C.sd, ref) %>%

filter(!is.na(mean), !is.na(std.err)) %>%

left\_join(ds[,c("study","agegroup", "tainf", "nlow")]) %>%

group\_by(study, subgroup, outcome, time) %>% mutate(sotid=cur\_group\_id()) %>% ungroup() %>%

arrange(sotid, ref, treatment)

## calculate SMD per study

for(g in unique(dn$sotid)){

ll <- covar.smd(mean, sd, n, measure="smd", method="hedges", data=filter(dn, sotid==g))

dn[dn$sotid==g, "diff"] <- c(NA, ll$y[-1])

#print(paste("sotid:", g," Diffs:", paste(c(NA, ll$y[-1]),collapse=",")))

if(length(ll$y)>2){

dn[dn$sotid==g, "diff\_se"] <- c(sqrt(1/dn$n[dn$sotid==g][1]), sqrt(ll$v)[-1])

}else{

dn[dn$sotid==g, "diff\_se"] <- c(NA, sqrt(ll$v)[-1])

}

}

# Explore average covariance before-after

# by outcome

# xx <- group\_by(dn, outcome) %>% summarise(cov=mean((B.se^2 + F.se^2 - C.se^2)/(2\*B.se\*F.se), na.rm=T), n=sum(!is.na(B.se + F.se - C.se)))

# Across outcomes

# mean((dn$B.se^2 + dn$F.se^2 - dn$C.se^2)/(2\*dn$B.se\*dn$F.se), na.rm=T)

## Prepare data for BINARY outcomes

db <- filter(do, !is.na(cases)) %>%

mutate(sampleSize = n,

responders = round(cases)) %>%

transmute(study, subgroup, treatment, outcome, time, weeks, sampleSize, responders) %>%

filter(!is.na(sampleSize), !is.na(responders)) %>%

left\_join(da[,c("study", "treatment", "ref")]) %>%

left\_join(ds[,c("study","agegroup", "tainf", "nlow")])

# table(db$outcome)

## Find subgroup of treatments disconnected from network

# xx <- transmute(dn, s.id=as.numeric(as.factor(study)), t.id=as.numeric(as.factor(treatment)), n, treatment, analysis)

# nma.networkplot(s.id, t.id, n, data=xx[xx$analysis %in% "itch",])

# levels(as.factor(xx$treatment))[c(7, 37, 38, 14)]

####################################

###

### Function to prepare the data for a particular meta-analysis

###

fpd <- function(DO, OUT, SMD, AGE=c("adult", "both"), TOP=NULL, NLR=NULL, TIM=NULL, REP=NULL, SUG=NULL){

# DO = dataset of outcomes

# OUT= set of outcomes to include by priority order

# SMD= (T/F) Standardised Mean Difference Analysis?

# AGE= all age groups to be included as in the trial tab> e.g. c("adult", "both")

# TOP= "y" topical treatment

# NLR= 7 number of low RoB items

# TIM= "s" time of follow up

# REP= c("c") kind of outcomce measure

# SUG= subgroup analysis

# select appropriate studies

if(!is.null(AGE)) DO <- filter(DO, tolower(agegroup) %in% tolower(AGE))

if(!is.null(TOP)) DO <- filter(DO, tolower(tainf) %in% tolower(TOP))

if(!is.null(NLR)) DO <- filter(DO, nlow %in% NLR)

if(!is.null(TIM)) DO <- filter(DO, tolower(time) %in% tolower(TIM))

if(!is.null(REP)) DO <- filter(DO, grepl(REP,rep))

if(!is.null(REP)) DO <- filter(DO, grepl(REP,rep))

if(!is.null(SUG)){

DO <- filter(DO, grepl(SUG, subgroup))

}else{

DO <- filter(DO, is.na(subgroup))

}

## Depending on type of analysis

if(SMD==T){

# Select variables depending on SMD change treatment name

DO <- mutate(DO, treatment=smd\_treat, std.err=diff\_se)

# Select appropriate outcomes

DO <- mutate(DO, priority=match(outcome, OUT)) %>% filter(!is.na(priority)) %>%

group\_by(study, time, ref, treatment) %>% summarize(minpri=min(priority), outcome=outcome[priority==minpri], n=n[priority==minpri], diff=diff[priority==minpri], std.err=std.err[priority==minpri]) %>% ungroup()

}else{

if("mean" %in% names(DO)){

DO <- mutate(DO, priority=match(outcome, OUT)) %>% filter(!is.na(priority)) %>%

group\_by(study, time, ref, treatment) %>% summarize(minpri=min(priority), outcome=outcome[priority==minpri], n=n[priority==minpri], mean=mean[priority==minpri], std.err=std.err[priority==minpri]) %>% ungroup()

}else{

DO <- mutate(DO, priority=match(outcome, OUT)) %>% filter(!is.na(priority)) %>%

group\_by(study, time, ref, treatment) %>% summarize(minpri=min(priority), outcome=outcome[priority==minpri], responders=responders[priority==minpri], sampleSize=sampleSize[priority==minpri]) %>% ungroup()

}

}

return(DO)

}

####################################

###

### Function to do simple meta-analysis of pair-wise comparisons

###

# Use data generated by funtion fpd()

fpwc <- function(DT, OUT, PTH, PRE, SMD=F, AGE=c("adult", "both"), TOP=NULL, NLR=NULL, TIM=NULL, REP=NULL, SUG=NULL){

# DO = dataset of outcomes

# OUT= set of outcomes to include by priority order

# PTH= pathway for storing results

# PRE= prefix for model name: e.g. "topic\_lowRoB\_short\_cfb"

# SMD= (T/F) Standardised Mean Difference Analysis?

# AGE= all age groups to be included as in the trial tab> e.g. c("adult", "both")

# TOP= "y" topical treatment

# NLR= 7 number of low RoB items

# TIM= "s" time of follow up

# REP= c("c") kind of outcomce measure

# SUG= subgroup analysis

if(dir.exists(PTH)==F) dir.create(PTH)

# select appropriate studies (but ignore SMD=T)

DO <- fpd(DO=DT, OUT=OUT, SMD=F, AGE=AGE, TOP=TOP, NLR=NLR, TIM=TIM, REP=REP, SUG=SUG)

print(head(DO))

if("mean" %in% names(DO)){

DO <- transmute(DO, studlab=study, treat=treatment, outcome, n, mean, sd=std.err\*sqrt(n))

}else{

DO <- transmute(DO, studlab=study, treat=treatment, outcome, n=sampleSize, event=responders)

}

tt <- sort(grep("Placebo", unique(DO$treat), value=T, invert=T))

tt <- c(tt, "Placebo")

while(length(tt)>1){

te <- tt[1]

tt <- tt[-1]

dd <- filter(DO, studlab %in% DO$studlab[DO$treat==te])

ot <- sort(unique(dd$treat)[unique(dd$treat) %in% tt])

for(tc in ot){

dw <- left\_join(filter(dd, treat==tc), filter(dd, treat==te), by=c("studlab","outcome"), suffix=c(".c",".e"))

if("mean" %in% names(DO)){

ma <- metacont(n.e, mean.e, sd.e, n.c, mean.c, sd.c, studlab, dw, sm=c("MD", "SMD")[SMD+1])

}else{

ma <- metabin(event.e, n.e, event.c, n.c, studlab, dw, sm="OR")

}

## Forest plot

pdf(file=paste0(PTH,"/",path," - ",te," - ",tc,PRE,".pdf"), width=15, height=2.5+(0.5\*nrow(dw)))

forest(ma, xlab=paste(te,"(Expe.) - ",tc,"(Cont.)"))

dev.off()

}

}

return(DO)

}

####################################

###

### Function to run the NMA

###

fnma <- function(DT, OUT, PTH, PRE, SMD=F, AGE=c("adult", "both"), TOP=NULL, NLR=NULL, TIM=NULL, REP=NULL, SUG=NULL, TMOD="consistency"){

# DO = dataset of outcomes

# OUT= set of outcomes to include by priority order

# PTH= pathway for storing results

# PRE= prefix for model name: e.g. "topic\_lowRoB\_short\_cfb"

# SMD= (T/F) Standardised Mean Difference Analysis?

# AGE= all age groups to be included as in the trial tab> e.g. c("adult", "both")

# TOP= "y" topical treatment

# NLR= 7 number of low RoB items

# TIM= "s" time of follow up

# REP= c("c") kind of outcome measure

# SUG= subgroup analysis

# TMOD = type pf model: "consistency" (classical) or "nodesplit"

if(dir.exists(PTH)==F) dir.create(PTH)

# select appropriate studies

DO <- fpd(DO=DT, OUT=OUT, SMD=SMD, AGE=AGE, TOP=TOP, NLR=NLR, TIM=TIM, REP=REP, SUG=SUG)

print(DO)

# Check number of disconected networks using function fnn

nn <- fnn(DO)

if(length(nn[["size"]])>1){

siz <- nn[["size"]]

t\_all <- nn[["t\_all"]]

t\_fin <- nn[[match(max(siz),siz)]]

t\_exc <- t\_all[!(t\_all %in% t\_fin)]

print(paste("There are",length(siz),"diconnected networks of treatments.\nThe treatments NOT included in the analysed network are:", paste(t\_exc, collapse=", ")))

write.csv(filter(DO, treatment %in% t\_exc), paste0(PTH,"/",PRE,"\_Excluded\_treatments.csv"))

DO <- filter(DO, treatment %in% t\_fin)

}

write.csv(DO, paste0(PTH,"/",PRE,"\_Data.csv"))

# build network

if(SMD==T){

net <- mtc.network(data.re=as.data.frame(DO))

}else{

net <- mtc.network(data.ab=as.data.frame(DO))

}

save(net, file=paste0(PTH,"/",PRE,"\_network.RData"))

# DEPENDING if NODE-SPLIT

if(TMOD=="nodesplit"){

mnc <- mtc.nodesplit.comparisons(net)

print(dim(mnc))

if(dim(mnc)[1]>0){

mod <- mtc.nodesplit(net, comparisons=mnc)

ns1s <- summary(mod)

pdf(file=paste0(PTH,"/",PRE,"\_NSplit.pdf"), width=10, height=10)

plot(ns1s)

dev.off()

topptx(plot(ns1s),file=paste0(PTH,"/",PRE,"\_NSplit.pptx"))

write.csv(as.data.frame(print(ns1s)), paste0(PTH,"/",PRE,"\_NSplit.csv"))

}else{

mss <- "There are no comparisons to assess for inconsistency. See https://doi.org/10.1002/jrsm.1167 for how this is determined"

print(mss)

cat(mss,file=paste0(PTH,"/",PRE,"\_NSplit.txt"))

}

}else{

#summary(net)

dev.off()

plot(net) # quick network plot of data - see if it makes sense before moving ahead

topptx(plot(net),file=paste0(PTH,"/",PRE,"\_network.pptx"))

## Define models depending if binary or continuous outcomes

if(SMD==F & ("responders" %in% names(DO))){

mod <- mtc.model(net, type=TMOD, hy.prior = mtc.hy.prior("var", "dlnorm", -2.99, 1/1.74^2), om.scale=(log(30)/1.96)^2/15)

}else{

mod <- mtc.model(net, type=TMOD)

}

save(mod,file=paste0(PTH,"/",PRE,"\_model.RData"))

# Update model

set.seed(1000)

res <- gemtc::mtc.run(mod, n.iter = 100000)

save(res, file=paste0(PTH,"/",PRE,"\_base.RData"))

# Other plots

pdf(file=paste0(PTH,"/",PRE,"\_diag1.pdf"))

gelman.plot(res)

dev.off()

pdf(file=paste0(PTH,"/",PRE,"\_diag2.pdf"))

plot(res)

dev.off()

# Calculate all pairwise comparisons

RET <- relative.effect.table(res)

# forest plots vs placebo

gemtc::forest(RET,"Placebo")

topptx(file=paste0(PTH,"/",PRE,"\_forest.pptx"), height=nrow(net$treatments)/1.5)

# ranks

ranks <- rank.probability(res,preferredDirection = -1)

max(nchar(rownames(ranks)))

par(mar=c(4,max(nchar(rownames(ranks)))-10,2,1), ask=F)

plot(ranks, beside=TRUE, las=1, xlim=c(0,max(ranks)), horiz=T)

topptx(file=paste0(PTH,"/",PRE,"\_ranks.pptx"))

# SUCRAs

sucras <- t(t(apply(apply(ranks,1,cumsum),2,mean)))

write.csv(sucras, paste0(PTH,"/",PRE,"\_SUCRAs.csv"))

# league tables

if(SMD==F & ("responders" %in% names(DO))){

ret <- round(relative.effect.table.lor.to.or(res),1)

}else{

ret <- round(RET,1)

}

write.csv(ret, paste0(PTH,"/",PRE,"\_league.csv"))

}

}

####################################

###

### Function to check the different networks of treatments in a set of trials

###

# data structure includes at least two columns, one for trial and another for treatment

# Use data generated by funtion fpd()

fnn <- function(DO){

# DO = data frame includes column for "study" and "treatment"

t\_all <- unique(DO$treatment)

ta <- t\_all

lnet <- list() # list of separated networks

s <- 0

while(length(ta)>0){

t1 <- sample(ta,1)

t0 <- NA

while(mean(t1 %in% t0)!=1){

t0 <- t1

t1 <- unique(c(t0, DO$treatment[DO$study %in% DO$study[DO$treatment %in% t0]]))

}

if(s==0) len <- length(t1) else len <- c(len, length(t1))

s <- s+1

lnet[[s]] <- t1

ta <- ta[!(ta %in% t1)]

}

lnet[["size"]] <- len

lnet[["t\_all"]] <- t\_all

return(lnet)

}

###

### Function from lor to or (writen by Alex)

###

relative.effect.table.lor.to.or <- function(result, covariate = NA) {

ts <- as.character(result[["model"]][["network"]][["treatments"]][["id"]])

tbl <- array(NA, dim = c(length(ts), length(ts), 3), dimnames = list(ts,

ts, c("2.5%", "50%", "97.5%")))

comps <- combn(ts, 2)

for (i in 1:ncol(comps)) {

comp <- comps[, i]

samples <- as.matrix(relative.effect(result, comp[1],

comp[2], preserve.extra = FALSE, covariate = covariate)$samples)

q <- quantile(exp(samples), prob = c(0.025, 0.5, 0.975))

tbl[comp[1], comp[2], ] <- unname(q)

q.inv <- c(1/q[3], 1/q[2], 1/q[1])

tbl[comp[2], comp[1], ] <- unname(q.inv)

}

attr(tbl, "model") <- result[["model"]]

attr(tbl, "covariate") <- covariate

class(tbl) <- "mtc.relative.effect.table"

tbl

}

###

### Function to ONLY generate network and node splitting

###

#

# fnmans <- function(DO, OUT, PTH, PRE, AGE=c("adult", "both"), TOP=NULL, NLR=NULL, TIM=NULL, REP=NULL){

# # DO = dataset of outcomes

# # PTH= pathway for storing results

# # PRE= prefix for model name: e.g. "topic\_lowRoB\_short\_cfb"

# # AGE= all age groups to be included as in the trial tab> e.g. c("adult", "both")

# # TOP= "y" topical treatment

# # NLR= 7 number of low RoB items

# # TIM= "s" time of follow up

# # REP= c("c") kind of outcomce measure

# if(dir.exists(PTH)==F) dir.create(PTH)

# # select appropriate studies

# if(!is.null(AGE)) DO <- filter(DO, tolower(agegroup) %in% tolower(AGE))

# if(!is.null(TOP)) DO <- filter(DO, tolower(tainf) %in% tolower(TOP))

# if(!is.null(NLR)) DO <- filter(DO, nlow %in% NLR)

# if(!is.null(TIM)) DO <- filter(DO, tolower(time) %in% tolower(TIM))

# if(!is.null(REP)) DO <- filter(DO, grepl(REP,rep))

# # Select appropriate outcomes

# DO <- filter(DO, analysis %in% OUT)

# # formulate the network give the above data

# net <- mtc.network(data.ab=as.data.frame(DO))

# #summary(net)

# ns1 <- mtc.nodesplit(net, comparisons=mtc.nodesplit.comparisons(net))

# ns1s <- summary(ns1)

# pdf(file=paste0(PTH,"/",PRE,"\_NSplit.pdf"), width=10, height=10)

# plot(ns1s)

# dev.off()

# topptx(plot(ns1s),file=paste0(PTH,"/",PRE,"\_NSplit.pptx"))

# write.csv(as.data.frame(print(ns1s)), paste0(PTH,"/",PRE,"\_NSplit.csv"))

# }

#

###

### End of functions

###

##############################3

###################

# LIBRARIES

###################

require(tidyverse)

require(gemtc)

require(eoffice)

require(pcnetmeta)

require(dosresmeta)

library(meta)

par(ask=F)

#myfile <- "AD-NMA-2022-10-04\_DP.xlsx"

myfile <- "AD-NMA-2022-12-09\_DP.xlsx"

source("AD-NMA\_read\_data4.R")

# Filter data

# Remove results in Adolescents

# for continuous outcomes

DN <- filter(dn, study != "Pacor Italy 2004", !grepl("Adolescents", study))

# For binary outcomes

DB <- filter(db, !grepl("Adolescents", study))

table(DB$outcome)

##################################.

####

#### For symptoms

####

outc <- "poem"

path <- "POEM"

outc <- arrange(filter(dc, !is.na(get(outc))), get(outc))$outcome

# Pair-wise comparison meta-analysis

fpwc(DN, outc, PTH="PAIRWISE", PRE="\_short", TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short"), TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_topical"), TIM="s", TOP="y", NLR=NULL, REP=NULL)

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_notopical"), TIM="s", TOP="n", NLR=NULL, REP=NULL)

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_lowRoB"), TIM="s", TOP=NULL, NLR=7, REP=NULL)

#### For node splitting ####.

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short"), TMOD="nodesplit", TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_topical"), TMOD="nodesplit", TIM="s", TOP="y", NLR=NULL, REP=NULL)

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_notopical"), TMOD="nodesplit", TIM="s", TOP="n", NLR=NULL, REP=NULL)

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_lowRoB"), TMOD="nodesplit", TIM="s", TOP=NULL, NLR=7, REP=NULL)

##################################.

####

#### For QoL

####

outc <- "qol"

path <- "QoL"

outc <- arrange(filter(dc, !is.na(get(outc))), get(outc))$outcome

# Pair-wise comparison meta-analysis

fpwc(DN, outc, PTH="PAIRWISE", PRE="\_short", TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short"), TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_cfb"), TIM="s", TOP=NULL, NLR=NULL, REP="c")

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_lowRoB"), TIM="s", TOP=NULL, NLR=7, REP=NULL)

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_lowRoB\_cfb"), TIM="s", TOP=NULL, NLR=7, REP="c")

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_topical"), TIM="s", TOP="y", NLR=NULL, REP=NULL)

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_topical\_cfb"), TIM="s", TOP="y", NLR=NULL, REP="c")

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_notopical"), TIM="s", TOP="n", NLR=NULL, REP=NULL)

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_notopical\_cfb"), TIM="s", TOP="n", NLR=NULL, REP="c")

#### For node splitting ####.

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short"), TMOD="nodesplit", TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_cfb"), TMOD="nodesplit", TIM="s", TOP=NULL, NLR=NULL, REP="c")

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_lowRoB"), TMOD="nodesplit", TIM="s", TOP=NULL, NLR=7, REP=NULL)

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_lowRoB\_cfb"), TMOD="nodesplit", TIM="s", TOP=NULL, NLR=7, REP="c")

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_topical"), TMOD="nodesplit", TIM="s", TOP="y", NLR=NULL, REP=NULL)

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_topical\_cfb"), TMOD="nodesplit", TIM="s", TOP="y", NLR=NULL, REP="c")

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_notopical"), TMOD="nodesplit", TIM="s", TOP="n", NLR=NULL, REP=NULL)

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_notopical\_cfb"), TMOD="nodesplit", TIM="s", TOP="n", NLR=NULL, REP="c")

##################################.

####

#### For Signs

####

outc <- "easi"

path <- "EASI"

outc <- arrange(filter(dc, !is.na(get(outc))), get(outc))$outcome

# Pair-wise comparison meta-analysis

fpwc(DN, outc, PTH="PAIRWISE", PRE="\_short", TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short"), TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_lowRoB"), TIM="s", TOP=NULL, NLR=7, REP=NULL)

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_notopical"), TIM="s", TOP="n", NLR=NULL, REP=NULL)

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_topical"), TIM="s", TOP="y", NLR=NULL, REP=NULL)

#### For node splitting ####.

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short"), TMOD="nodesplit", TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_lowRoB"), TMOD="nodesplit", TIM="s", TOP=NULL, NLR=7, REP=NULL)

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_notopical"), TMOD="nodesplit", TIM="s", TOP="n", NLR=NULL, REP=NULL)

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_topical"), TMOD="nodesplit", TIM="s", TOP="y", NLR=NULL, REP=NULL)

##################################.

####

#### For Itch

####

outc <- "itch"

path <- "Itch"

outc <- arrange(filter(dc, !is.na(get(outc))), get(outc))$outcome

# Pair-wise comparison meta-analysis

fpwc(DN, outc, PTH="PAIRWISE", PRE="\_short", TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short"), TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_cfb"), TIM="s", TOP=NULL, NLR=NULL, REP="c")

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_lowRoB"), TIM="s", TOP=NULL, NLR=7, REP=NULL)

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_notopical"), TIM="s", TOP="n", NLR=NULL, REP=NULL)

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_topical"), TIM="s", TOP="y", NLR=NULL, REP=NULL)

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_lowRoB\_cfb"), TIM="s", TOP=NULL, NLR=7, REP="c")

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_topical\_cfb"), TIM="s", TOP="y", NLR=NULL, REP="c")

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_notopical\_cfb"), TIM="s", TOP="n", NLR=NULL, REP="c")

#### For node splitting ####.

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short"), TMOD="nodesplit", TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_cfb"), TMOD="nodesplit", TIM="s", TOP=NULL, NLR=NULL, REP="c")

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_lowRoB"), TMOD="nodesplit", TIM="s", TOP=NULL, NLR=7, REP=NULL)

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_notopical"), TMOD="nodesplit", TIM="s", TOP="n", NLR=NULL, REP=NULL)

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_topical"), TMOD="nodesplit", TIM="s", TOP="y", NLR=NULL, REP=NULL)

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_lowRoB\_cfb"), TMOD="nodesplit", TIM="s", TOP=NULL, NLR=7, REP="c")

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_topical\_cfb"), TMOD="nodesplit", TIM="s", TOP="y", NLR=NULL, REP="c")

fnma(DN, outc, PTH=path, PRE=paste0(path,"\_short\_notopical\_cfb"), TMOD="nodesplit", TIM="s", TOP="n", NLR=NULL, REP="c")

##################################.

####

#### For SAE

####

path <- "SAE"

outc <- "sae"

outc <- arrange(filter(dc, !is.na(get(outc))), get(outc))$outcome

fpwc(DB, outc, PTH="PAIRWISE", PRE="\_short", TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short"), TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_lowRoB"), TIM="s", TOP=NULL, NLR=7, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_notopical"), TIM="s", TOP="n", NLR=NULL, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_topical"), TIM="s", TOP="y", NLR=NULL, REP=NULL)

#### For node splitting ####.

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short"), TMOD="nodesplit", TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_lowRoB"), TMOD="nodesplit", TIM="s", TOP=NULL, NLR=7, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_notopical"), TMOD="nodesplit", TIM="s", TOP="n", NLR=NULL, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_topical"), TMOD="nodesplit", TIM="s", TOP="y", NLR=NULL, REP=NULL)

##################################.

####

#### For WD

####

path <- "WD"

outc <- "wdr"

outc <- arrange(filter(dc, !is.na(get(outc))), get(outc))$outcome

fpwc(DB, outc, PTH="PAIRWISE", PRE="\_short", TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short"), TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_lowRoB"), TIM="s", TOP=NULL, NLR=7, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_notopical"), TIM="s", TOP="n", NLR=NULL, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_topical"), TIM="s", TOP="y", NLR=NULL, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_adult"), AGE=c("adult"), TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

#### For node splitting ####.

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short"), TMOD="nodesplit", TIM="s", TOP=NULL, NLR=NULL, REP=NULL) #

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_lowRoB"), TMOD="nodesplit", TIM="s", TOP=NULL, NLR=7, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_notopical"), TMOD="nodesplit", TIM="s", TOP="n", NLR=NULL, REP=NULL) #

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_topical"), TMOD="nodesplit", TIM="s", TOP="y", NLR=NULL, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_adult"), AGE=c("adult"), TMOD="nodesplit", TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

##################################.

####

#### For IGA

####

path <- "IGA"

outc <- "iga"

outc <- arrange(filter(dc, !is.na(get(outc))), get(outc))$outcome

fpwc(DB, outc, PTH="PAIRWISE", PRE="\_short", TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short"), TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short"), TIM="s", TOP=NULL, NLR=NULL, REP=NULL, SUG="adults")

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_lowRoB"), TIM="s", TOP=NULL, NLR=7, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_notopical"), TIM="s", TOP="n", NLR=NULL, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_topical"), TIM="s", TOP="y", NLR=NULL, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_adult"), AGE=c("adult"), TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

#### For node splitting ####.

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short"), TMOD="nodesplit", TIM="s", TOP=NULL, NLR=NULL, REP=NULL) #

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_lowRoB"), TMOD="nodesplit", TIM="s", TOP=NULL, NLR=7, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_notopical"), TMOD="nodesplit", TIM="s", TOP="n", NLR=NULL, REP=NULL) #

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_topical"), TMOD="nodesplit", TIM="s", TOP="y", NLR=NULL, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_adult"), AGE=c("adult"), TMOD="nodesplit", TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

##################################.

####

#### For EASI50

####

path <- "EASI50"

outc <- "easi50"

outc <- arrange(filter(dc, !is.na(get(outc))), get(outc))$outcome

fpwc(DB, outc, PTH="PAIRWISE", PRE="\_short", TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short"), TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_lowRoB"), TIM="s", TOP=NULL, NLR=7, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_notopical"), TIM="s", TOP="n", NLR=NULL, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_topical"), TIM="s", TOP="y", NLR=NULL, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_adult"), AGE=c("adult"), TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

#### For node splitting ####.

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short"), TMOD="nodesplit", TIM="s", TOP=NULL, NLR=NULL, REP=NULL) #

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_lowRoB"), TMOD="nodesplit", TIM="s", TOP=NULL, NLR=7, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_notopical"), TMOD="nodesplit", TIM="s", TOP="n", NLR=NULL, REP=NULL) #

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_topical"), TMOD="nodesplit", TIM="s", TOP="y", NLR=NULL, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_adult"), AGE=c("adult"), TMOD="nodesplit", TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

##################################.

####

#### For EASI75

####

path <- "EASI75"

outc <- "easi75"

outc <- arrange(filter(dc, !is.na(get(outc))), get(outc))$outcome

fpwc(DB, outc, PTH="PAIRWISE", PRE="\_short", TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short"), TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_lowRoB"), TIM="s", TOP=NULL, NLR=7, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_notopical"), TIM="s", TOP="n", NLR=NULL, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_topical"), TIM="s", TOP="y", NLR=NULL, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_adult"), AGE=c("adult"), TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

#### For node splitting ####.

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short"), TMOD="nodesplit", TIM="s", TOP=NULL, NLR=NULL, REP=NULL) #

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_lowRoB"), TMOD="nodesplit", TIM="s", TOP=NULL, NLR=7, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_notopical"), TMOD="nodesplit", TIM="s", TOP="n", NLR=NULL, REP=NULL) #

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_topical"), TMOD="nodesplit", TIM="s", TOP="y", NLR=NULL, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_adult"), AGE=c("adult"), TMOD="nodesplit", TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

##################################.

####

#### For EASI90

####

path <- "EASI90"

outc <- "easi90"

outc <- arrange(filter(dc, !is.na(get(outc))), get(outc))$outcome

fpwc(DB, outc, PTH="PAIRWISE", PRE="\_short", TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short"), TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_lowRoB"), TIM="s", TOP=NULL, NLR=7, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_notopical"), TIM="s", TOP="n", NLR=NULL, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_topical"), TIM="s", TOP="y", NLR=NULL, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_adult"), AGE=c("adult"), TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

#### For node splitting ####.

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short"), TMOD="nodesplit", TIM="s", TOP=NULL, NLR=NULL, REP=NULL) #

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_lowRoB"), TMOD="nodesplit", TIM="s", TOP=NULL, NLR=7, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_notopical"), TMOD="nodesplit", TIM="s", TOP="n", NLR=NULL, REP=NULL) #

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_topical"), TMOD="nodesplit", TIM="s", TOP="y", NLR=NULL, REP=NULL)

fnma(DB, outc, PTH=path, PRE=paste0(path,"\_short\_adult"), AGE=c("adult"), TMOD="nodesplit", TIM="s", TOP=NULL, NLR=NULL, REP=NULL)

##################################.

####

#### For SMD

####

# read all SMD domains

smds <- grep("SMD", names(dc), value=T)

# for each domain run the analysis

for(v in smds[-1]){

outc <- arrange(filter(dc, !is.na(get(v))), get(v))$outcome

fnma(DN, outc, PTH=v, PRE=paste0(v,"\_short"), SMD=T, TIM="s")

fnma(DN, outc, PTH=v, PRE=paste0(v,"\_short"), SMD=T, TIM="s", TMOD="nodesplit")

}

## Pairwise analysis

for(v in smds[-1]){

path <- v

outc <- arrange(filter(dc, !is.na(get(v))), get(v))$outcome

fpwc(DN, outc, PTH="PAIRWISE", PRE="\_short", TIM="s", SMD=T)

}