#'@name grabclean

#'@title Cleaning grab data

#'@description if column names do not match up make sure there is only one "date" column

#'@param yearmon numeric survey date in yyyymm format

#'@param tofile logical save output to file

#'@param fdir character file path to local data directory

#'@details If streaming data does not exist for a particular data/time pull averages for the previous minute (if data exists).

#'@export

#'@examples \dontrun{

#'res <- grabclean(yearmon = 201402, tofile = FALSE)

#'res <- grabclean(yearmon = 200808, tofile = FALSE)

#'res <- grabclean(yearmon = 201212, tofile = FALSE)

#'res <- grabclean(yearmon = 201007, tofile = FALSE)

#'res <- grabclean(yearmon = 201004, tofile = FALSE)

#'}

#'

grabclean <- function(yearmon, tofile = FALSE, fdir = getOption("fdir")){

formatcolnames <- function(sumpath){

datacol <- NA

nmsfull <- NA

startread <- 1

endread <- 23

#READ LABID thru PP

nms1 <- read.csv(sumpath, sep = ",", skip = 2, header = F, stringsAsFactors = T, na.strings = "", strip.white = T)[1, startread:endread]

nms1 <- apply(nms1, 2, as.character)

while(nms1[length(nms1)] != "PP"){

endread <- endread - 1

nms1 <- read.csv(sumpath, sep = ",", skip = 2, header = F, stringsAsFactors = T, na.strings = "", strip.white = T)[1, startread:endread]

nms1 <- apply(nms1, 2, as.character)

}

datacol <- startread:endread

#READ NITROGEN AND PHOSPHORUS COLUMNS (SERC NUTRIENTS)

startread <- endread + 1

endread <- startread + 8

#if(any(nchar(nms1)>10)){nms1<-nms1[-which(nchar(nms1)>10)]}

nms2 <- read.csv(sumpath, sep = ",", skip = 3, header = F, stringsAsFactors = T, na.strings = "",strip.white = T)

#nms2<-nms2[1,25:28]

nms2 <- nms2[1,startread:endread]

nms2 <- apply(nms2,2,as.character)

while(nms2[length(nms2)] != "SRP(uM)"){

endread <- endread - 1

nms2 <- read.csv(sumpath, sep = ",", skip = 3, header = F, stringsAsFactors = T, na.strings = "", strip.white = T)

#nms2<-nms2[1,25:28]

nms2 <- nms2[1, startread:endread]

nms2 <- apply(nms2, 2, as.character)

}

while(gsub(" ", "", nms2[1]) != "N+N(uM)"){

nms2 <- nms2[-1]

startread <- startread + 1

}

datacol <- append(datacol, (startread:endread))

#CNP DATA (DISTRICT LAB)

startread <- endread + 1

endread <- startread + 9

nms3 <- read.csv(sumpath, sep = ",", skip = 2, header = F, stringsAsFactors = T, na.strings = "", strip.white = T)

#nms3<-nms3[1,29:38]

#nms3<-nms3[1,32:38]

nms3 <- nms3[1, startread:endread]

nms3 <- apply(nms3, 2, as.character)

while(nms3[length(nms3)] != "TDKN"){

endread <- endread - 1

nms3 <- read.csv(sumpath, sep = ",", skip = 2, header = F, stringsAsFactors = T, na.strings = "", strip.white = T)

#nms2<-nms2[1,25:28]

nms3 <- nms3[1, startread:endread]

nms3 <- apply(nms3, 2, as.character)

}

if(is.na(nms3[1])){

nms3 <- nms3[-1]

startread <- startread + 1

}

while(is.na(nms3[1]) | nms3[1] != "TP"){

nms3 <- nms3[-1]

startread <- startread + 1

}

datacol <- append(datacol, (startread:endread))

#EXISTING STREAMING DATA

startread <- endread + 1

nms4 <- read.csv(sumpath, sep = ",", skip = 1, header = F, stringsAsFactors = T, na.strings = "", strip.white = T)

if(ncol(nms4) > endread){#is there C6 data?

nms4 <- nms4[1, startread:ncol(nms4)]

nms4 <- apply(nms4, 2, as.character)

nms.full <- c(nms1, nms2, nms3, nms4)

datacol <- append(datacol, (startread:(startread + length(nms4) - 1)))

}else{

nms.full <- c(nms1, nms2, nms3)

nms.full <- c(nms.full, rep(NA, (77 - length(nms.full))))#need to fix this

}

nms.full <- tolower(nms.full)

nms.full <- gsub(" ", "", nms.full)

nms.full <- gsub("\\(", "", nms.full)

nms.full <- gsub(")", "", nms.full)

if(length(which(is.na(nms.full))) > 0){

datacol <- datacol[-which(is.na(nms.full))]

nms.full <- nms.full[!is.na(nms.full)]

}

list(nms.full, datacol)

}

cleangrabdata <- function(sumpath, nsmfull, datacol){

grabdata <- read.csv(sumpath, sep = ",", skip = 5, header = FALSE, stringsAsFactors = FALSE, na.strings = "", strip.white = TRUE)

grabdata <- grabdata[!is.na(grabdata[,5]) | !is.na(grabdata[,4]),]#remove trailing blank rows

grabdata <- grabdata[,datacol]

names(grabdata) <- nmsfull

grabdata <- grabdata[,colSums(is.na(grabdata)) < nrow(grabdata)]#remove columns of all NA

narows <- as.numeric(which(apply(grabdata, 1, function(x)sum(is.na(x))) > 50))

if(length(narows) > 0){

grabdata <- grabdata[-unique(narows),]#remove NA rows

}

grabdata <- grabdata[!is.na(grabdata[,"chla"]),] #eliminate EBs

#FORMAT DATES

grabdata[,4] <- gsub("/", "", grabdata[,4])

grabdata[,4] <- gsub("-", "", grabdata[,4])

grabdata <- grabdata[,!is.na(names(grabdata))]

#EXTRACT UNIQUE IDS (stations) AS DATE+TIME

if(nchar(as.character(grabdata[1,5])) < 5){

names(grabdata)[4:5] <- c("date", "time")#make sure to match parameter names for stations and dt

grabdata[,4:5] <- apply(grabdata[,4:5], 2, as.numeric)

stations <- grabdata[,4:5]

}else{

names(grabdata)[5:6] <- c("date", "time")#make sure to match parameter names for stations and dt

grabdata[,5:6] <- apply(grabdata[,5:6], 2, as.numeric)

stations <- grabdata[,5:6]

}

stations <- data.frame(apply(stations, 2, function(x) as.numeric(as.character(x))))

sixchardate <- function(x){

yr <- sapply(x, function(x) substring(x, nchar(x) - 1, nchar(x)))

mon <- substring(x, 1, 2)

if(mean(as.numeric(mon)) > 12){

mon <- paste("0", substring(x, 1, 1), sep = "")

day <- sapply(x,function(x)

if(substring(x, 2, 3) > 31){

paste("0", substring(x, 2, 2), sep = "")

}else{

substring(x, 2, 3)

})

}else{

day <- paste("0", substring(x, 3, 3), sep = "")

}

paste(mon, day, yr, sep = "")

}

dlen <- mean(nchar(stations$date))

if(dlen != 6){

stations$date <- sixchardate(stations$date)

}

# dlen<-mean(nchar(as.character(streamingdata$date)))

# if(dlen!=6){

# print("1")

# if(!identical(as.character(streamingdata$date),gsub("/","",streamingdata$date))){

# print("2")

# streamingdata$date<-gsub("/","",streamingdata$date)

# if(dlen>8){

# print("3")

# streamingdata$date<-paste(substring(as.character(streamingdata$date),1,4),substring(as.character(streamingdata$date),7,8),sep="")

# }

# }else{

# print("4")

# streamingdata$date<-sixchardate(streamingdata$date)

# }

# }

#clear any streaming data already entered

if(any(names(grabdata) == "chlaiv")){

grabdata <- grabdata[,1:(which(names(grabdata) == "chlaiv") - 1)]

}

list(grabdata = grabdata, stations = stations)

}

#CALCULATE STREAMING AVERAGES CORRESPONDING TO GRABS

mergegrabstreaming <- function(streamingdata, grabdata, stations){

names(streamingdata)[names(streamingdata) == "chla"] <- "chlaiv"

stream <- merge(stations, streamingdata)#cuts dt down to match "stations"

if(nrow(stream) == 0){#streamingdata has 5 character dates

stations$date <- substring(stations$date, 2,

mean(nchar(stations$date)))

stream <- merge(stations,streamingdata)

}

if(nrow(stream) == 0){#streamingdata has dates with "/" character

streamingdata$date <- as.numeric(paste0(

strftime(streamingdata$datetime, format = "%m"),

strftime(streamingdata$datetime, format = "%d"),

substring(strftime(streamingdata$datetime, format = "%Y"), 3, 4)

))

stream <- merge(stations,streamingdata)

}

if(nrow(stream) == 0){ #streaming data has 8 character times

streamingdata[,"time"] <- as.numeric(gsub(":", "", substr(streamingdata[,"time"], 1, 5)))

stream <- merge(stations,streamingdata)

}

#check to make sure streaming data exists for each grab

nostream <- data.frame(matrix(NA, nrow = 1, ncol = ncol(grabdata)))

names(nostream) <- names(grabdata)

cnt <- 0

#change to create a vector of lines to remove rather than updating within loop

rmlist <- list()

for(j in 1:nrow(stations)){

if(all(is.na(match(paste(stream$date, stream$time), paste(stations[j,1], stations[j,2]))))){

if(cnt == 0){

nostream[1,] <- grabdata[j,]

cnt <- 1

}else{

nostream <- rbind(nostream, grabdata[j,])

}

rmlist[[j]] <- j

}

}

#check if streaming data exists for the minute previous to nostream

for(k in 1:nrow(nostream)){

nostreamprevious <- streamingdata[

paste(streamingdata[,"date"],streamingdata[,"time"])

==

paste(nostream[k, "date"], (nostream[k,"time"] - 1))

,]

nostreamprevious[,"time"] <- nostreamprevious[,"time"] + 1

if(nrow(nostreamprevious) > 0){

stream <- rbind(stream, nostreamprevious)

grabdata[paste(grabdata[,"date"], grabdata[,"time"]) == paste(nostream[k, "date"], nostream[k, "time"]),]$time <- grabdata[paste(grabdata[,"date"], grabdata[,"time"]) == paste(nostream[k, "date"], nostream[k, "time"]),]$time + 1

nostream <- nostream[-k,]

rmlist <- unlist(rmlist)[-k]

}

}

if(length(unlist(rmlist)) > 0){

stations <- stations[-unlist(rmlist),]

grabdata <- grabdata[-unlist(rmlist),]

}

#identical(stationsave,stations)#should be false

#make sure nrow stations and nrow stream2 match

stream$date <- as.numeric(stream$date)

stream <- merge(stations, stream)

stream2 <- data.frame(matrix(NA, nrow = nrow(stations), ncol = ncol(stream)))

for(m in 1:ncol(stream)){

if(class(stream[,m]) == "numeric"){

stream2[,m] <- round(aggregate(stream[,m], by = list(stream$date, stream$time), mean)[,3], 5)

}else{

stream2[,m] <- aggregate(stream[,m], by = list(stream$date, stream$time), Mode)[,3]

}

}

names(stream2) <- names(stream)

stream3 <- stream2[order(stream2$date, stream2$time),]

# sname <- which(names(stream3) == "chlaiv")

# ename <- which(names(stream3) == "lat\_dd")

stream4 <- cbind(stations, stream3)

grabsfull <- cbind(grabdata, stream4)

#add back in grabs with missing streaming data

if(any(!is.na(nostream[1,]))){

nostream <- nostream[!is.na(nostream[,4]),]

if((ncol(grabsfull) - ncol(nostream)) != 0){

padna <- data.frame(matrix(NA, nrow = nrow(nostream), ncol = (ncol(grabsfull) - ncol(nostream))))

#if(any(match(names(nostream),names(grabsfull))[1:ncol(nostream)]!=1:ncol(nostream))){

# stop("problem with column names")

#}

nostream <- cbind(nostream, padna)

}

names(nostream) <- names(grabsfull)

test <- rbind(grabsfull[1,], nostream)

grabsfull <- rbind(grabsfull, nostream)

}

namestemp <- c("date", "time", "location", "salt", "chla", "chla.1", "tss", "tss.1", "n.num", "no3um", "no2um", "nh4um", "tinum", "srpum", "pp", "pp.1", "tp", "tdp", "po4", "toc", "doc", "tkn", "tdkn", "temp.deg.c", "turb.ntu", "ph.units", "spcond.ms.cm", "chl.ug.l", "salinity.pss", "hdo.mg.l", "hdo..sat", "brighteners", "phycoe", "phycoc", "c6chl", "c6cdom", "c6turbidity", "c6temp", "lon\_dd", "lat\_dd")

nseq <- seq(1, length(namestemp), 1)

namesalias <- read.table(text = "chlorophyll.a,c6chl

c6chla,c6chl

spcondms,spcond

turbidity,c6turbidity

n+num,n.num", sep = ",")

namesalias <- apply(namesalias, 2, function(x) as.character(x))

#match dt names to a template that includes all possible columns####

for(n in 1:ncol(grabsfull)){

if(any(names(grabsfull)[n] == namesalias[,1])){

names(grabsfull)[n] <- namesalias[which(names(grabsfull)[n] == namesalias[,1]), 2]

}

}

#trim extra columns and match order to template

nmiss <- nseq[!(nseq %in% match(names(grabsfull), namestemp))]

if(length(nmiss) > 0){

for(j in 1:length(nmiss)){

grabsfull[,ncol(grabsfull) + 1] <- NA

names(grabsfull)[ncol(grabsfull)] <- namestemp[nmiss[j]]

}

}

grabsfull <- grabsfull[,match(namestemp, names(grabsfull))]#sort to match order of namestemp

grabsfull[,5:ncol(grabsfull)] <- suppressWarnings(apply(grabsfull[,5:ncol(grabsfull)], 2, function(x) as.numeric(x)))

grabsfull$flags <- NA

grabsfull

}

consistentlocations <- function(dt){

fathombasins <- rgdal::readOGR(file.path(fdir, "DF\_Basefile/fbzonesmerge.shp"), layer = "fbzonesmerge", verbose = FALSE)

#slot(fathombasins, "data")

nonna\_dt\_names <- which(!is.na(dt[,"lon\_dd"]) & !is.na(dt[,"lat\_dd"]))

dt\_over <- coordinatize(dt, latname = "lat\_dd", lonname = "lon\_dd")

dt\_over <- sp::over(dt\_over, fathombasins)

res <- as.character(dt\_over$ZoneName)

res[which(is.na(res))] <- dt[nonna\_dt\_names,]$location[which(is.na(res))]

dt[!is.na(dt[,"lon\_dd"]) & !is.na(dt[,"lat\_dd"]),]$location <- res

dt$location

}

#EXECUTION BLOCK####

#load files####

fdir\_fd <- file.path(fdir,"DF\_FullDataSets")

flist <- list.files(fdir\_fd, include.dirs = T, full.names = T)

streamingdata <- streamget(yearmon)

fdir\_fd <- file.path(fdir, "DF\_GrabSamples", "Raw")

flist <- list.files(fdir\_fd, include.dirs = T, full.names = T, pattern = ".csv")

sumpath <- suppressWarnings(flist[which(as.numeric(substring(basename(flist), 1, 6)) == yearmon)])

#todo: add check that sumpath only returns one file path

#CLEAN AND AGGREGRATE

grabnames <- formatcolnames(sumpath)

nmsfull <- unlist(grabnames[1])

datacol <- unlist(grabnames[2])

grabdata <- cleangrabdata(sumpath, nmsfull, datacol)$grabdata

stations <- cleangrabdata(sumpath, nmsfull, datacol)$stations

grabsfull <- mergegrabstreaming(streamingdata, grabdata, stations)

grabsfull$location <- consistentlocations(grabsfull)

if(tofile == TRUE){

write.csv(grabsfull, file.path(fdir, "DF\_GrabSamples", paste(yearmon, "j.csv", sep = "")))

}

return(grabsfull)

}