Função idarea

funcao.csv

```
idarea <- function(df) {
  #Verificando os parametros
  if(class(df)!="data.frame") #se o arquivo não for data.frame
    stop("Arquivo não é um data.frame") #retorna esta mensagem
  if(df!=(c(df$Ptt,df$Date, df$Latitude, df$Longitude, df$Quality))) #se o
arquivo não tiver as colunas Ptt, Date, Latitude, Longitude
    stop("Arquivo não possui as colunas necessárias") #retorna esta mensagem
  #Acessando pacotes necessários
  require(tidyverse)
  require(lubridate)
  require(rgdal)
  require(sp)
  require(dplyr)
  df<- filter(df, Quality %in% c("1", "2", "3","A","B")) #tirando posição Z
  #convertendo a coluna Date em as.POSIxct e criando uma nova coluna só com
dia/mes/ano
  df$Date<- as.character(df$Date) #convertendo a coluna Date em character
  df$Days <- sapply(strsplit(df$Date, split=' ', fixed=TRUE), function(x)
(x[2])) #separando a coluna Date (ano/mes/dia hora/min/seg) em uma nova
coluna Days (ano/mes/dia)
  df$Days <- as.POSIXct(df$Days, format = "%d-%b-%Y",tz = "GMT",usetz =
TRUE) #convertendo a coluna Days em as.POSIXct
  df$Date <- as.POSIXct(df$Date, format = "%H:%M:%S %d-%b-%Y",tz =
"GMT",usetz = TRUE) #convertendo a coluna Date em as.POSIXct
  #criando coluna 0-40 dias e 41-x dias desde a primeira transmissão do TAG
de cada animal----
  #Past_Thresdold 0 e 1
  df<-df%>% #chama o df
    group_by(Ptt) %>% #pelo grupo/coluna Ptt
    mutate(Threshold = min(Days) + days(40)) %>% #cria coluna Threshold
pegando a menor data de cada Ptt e soma 40 dias
    ungroup() %>% #retira o "grupo"
    mutate(Past_Threshold = Days > Threshold) #cria coluna Past_Threshold,
TRUE (>40 dias) e FALSE (0 <=40 dias)
  df$Past_Threshold<-as.numeric(df$Past_Threshold) #transforma os valores
TRUE e FALSE em numeros 0-1
  #0 False(0 < = 40), 1 True ( > 40)
  ang.abs <- function(x) {#calculando o angulo absoluto entre Lat e Long
    x1 <- x[-1, ]
    x2 <- x[-nrow(x), ]
    dist <- c(sqrt((x1$Latitude - x2$Latitude)^2 + (x1$Longitude -
x2$Longitude)^2), NA)#calculando a distância entre as localizações, se não
tiver pelo menos 2 localizações sucessivas retorna NA
    R2n <- (x$Latitude - x$Latitude[1])^2 + (x$Longitude - x$Longitude[1])^2
#calculando o valor de deslocamento bruto entre a relocação atual e a
primeira relocação da rota
```

```
    dt <- c(unclass(x1$Date) - unclass(x2$Date), NA)#calculando o tempo
decorrido entre as localizações, se nao tiver datas sucessivas, retorna NA
    dx <- c(x1$Latitude - x2$Latitude, NA)#calculando o valor do movimento
na direção x entre as localizações, se nao tiver pelo menos 2 localizações
sucessivas, retorna NA
    dy <- c(x1$Longitude - x2$Longitude, NA)#calculando o valor do movimento
na direção y entre as localizações, se nao tiver pelo menos 2 localizações
sucessivas, retorna NA
    abs.angle <- ifelse(dist < 1e-07, NA, atan2(dy, dx))#calculando o angulo
entre os movimentos pelas localizações,  se nao tiver pelo menos 2
localizações sucessivas, retorna NA
    so <- cbind.data.frame(dx = dx, dy = dy, dist = dist,
                           dt = dt, R2n = R2n, abs.angle = abs.angle)
#juntando todos os objetos criados em um data.frame chamado so
  }
  df1<- cbind(df,ang.abs(df)) #juntando o df com o resultado do angulo
absoluto
  ang.rel <- function(df,  slsp = c("remove",
                                    "missing")) {#calculando o angulo
relativo, slsp controla os valores retornados para realocações (se tiver
localizações em um mesmo local), missing: um valor ausente é retornado para
determinada realocação, remove: calcula o angulo entre as localizações
próximas diferentes
    ang1 <- df1$abs.angle[-nrow(df1)]#cria ang1, onde é a coluna abs.angle
menos a primeira linha sucessivament
    ang2 <- df1$abs.angle[-1]# cria ang2, onde é a coluna abs.angle menos 1
e sucessivamente
    slsp <- match.arg(slsp)#conferindo combinações de valores slsp
    if (slsp == "remove") {#se slsp for escolhido como remove
      dist <- c(sqrt((df1[-nrow(df1), "Latitude"] - df[-1, "Latitude"])^2 +
                       (df1[-nrow(df1), "Longitude"] - df[-1,
"Longitude"])^2), NA) #calcula a distancia
      wh.na <- which(dist < 1e-07) #cria objeto wh.na onde a distancia for
menor que 1e-07
      if (length(wh.na) > 0) {#se o comprimento de wh.na for menor que 0
        no.na <- (1:length(ang1))[!(1:length(ang1)) %in% #cria o objeto
no.na
                                  wh.na]
        for (i in wh.na) {#então roda por wh.na
          indx <- no.na[no.na < i]#criando o objeto indx, com valores no.na
menores que i ao longo que for rodado
          ang1[i] <- ifelse(length(indx) == 0, NA, ang1[max(indx)])#cria o
objeto ang1, onde se o comprimento de indx for TRUE para valor 0, retorne
NA, caso for FALSE retorne o valor máximo do ang1
        }
      }
    }
    res <- ang2 - ang1#subtraindo objeto ang2 por ang1 e resultando em res
    res <- ifelse(res <= (-pi), 2 * pi + res, res)#se res for TRUE para
menor ou igual a - pi, retorne 2*pi+res, FALSE retorne res
```

```
    res <- ifelse(res > pi, res - 2 * pi, res)#se res for TRUE para maior
que pi, retorne res - 2*pi, se FALSE retorne res
    return(c(NA, res))#retornando res
  }
  rel.angle<-abs(ang.rel(df1)) #tornando valores absolutos
  rel.angle[is.na(rel.angle)] <- 0 #substituindo NA por 0
  rel <- cbind.data.frame(rel.angle=rel.angle)#juntando ang. relativo e ang.
absoluto em rel, tornando-o em data.frame
  df2<-cbind(df1,rel) #juntando df1 com o rel
  categ <- function(df)
  {
    #criando threshold dos angulos
    df2$thr_angle[df2$rel.angle >= 1.01] <- "a" #AR e AA, valores próximos
de 1, rota SINUOSA
    df2$thr_angle[df2$rel.angle <=1.00] <- "b" #AM, valores próximos de 0,
rota RETILINEA
    df2$thr_angle[is.na(df2$thr_angle)] <- "a"  #substituindo NA por 0
    latcat<-cut(df2$Latitude, c(-Inf,-40,Inf),labels=c(1,0))#criando limite
-40.0000 na latitude
    cod<-paste(df2$Past_Threshold , df2$thr_angle, latcat, sep="-")#juntando
a coluna Past_Threshold e thr_angle em uma nova coluna chamada cod, com
separação "-"
    ct <- cbind.data.frame(latcat=latcat,cod=cod)#juntando o limite de
latitude latcat com a coluna cod em um data.frame chamado ct
  }
  df3<-cbind(df2,as.data.frame(categ(df2)))#tornando o df2 em data.frame
chamado df3
  if(df3$Latitude >= -22.000){ #se a latitude da primeira posição for até
-22.000 o transmissor foi implantado na AR
    #criando colunas AM, AA, AR de acordo com Past_Threshold (1-0),
thr_angle(a-b) e limite de latitude (0-1).
    df3$area[df3$cod == "1-a-0"] <- "AM"#criando a coluna AM (area de
migração)
    df3$area[df3$cod == "1-a-1"] <- "AA"#criando a coluna AA (area de
alimentação)
    df3$area[df3$cod == "1-b-0"] <- "AM"#criando a coluna AM (area de
migração)
    df3$area[df3$cod == "1-b-1"] <- "AM"#criando a coluna AM (area de
migração)
    df3$area[df3$cod == "0-a-0"] <- "AR"#criando a coluna AR (area de
reprodução)
    df3$area[df3$cod == "0-b-0"] <- "AR"#criando a coluna AR (area de
reprodução)
  }
  if(df3$Latitude <=-45.000){#se a latitude da primeira posição for além de
-45.000 o transmissor foi implantado na AA
    df3$area[df3$cod == "1-a-0"] <- "AR"#criando a coluna AR (area de
reprodução)
    df3$area[df3$cod == "1-a-1"] <- "AA"#criando a coluna AA (area de
alimentação)
    df3$area[df3$cod == "1-b-0"] <- "AM"#criando a coluna AM (area de
```

```
migração)
    df3$area[df3$cod == "1-b-1"] <- "AA"#criando a coluna AA (area de
alimentação)
    df3$area[df3$cod == "0-a-0"] <- "AM"#criando a coluna AM (area de
migração)
    df3$area[df3$cod == "0-b-0"] <- "AR"#criando a coluna AR (area de
reprodução)
  }
  df3$dt = NULL; df3$dx =NULL; df3$dy = NULL; df3$dist = NULL ; df3$latcat =
NULL; df3$cod = NULL; df3$R2n = NULL; df3$rel.angle = NULL; df3$abs.angle =
NULL ; df3$Days = NULL; df3$Past_Threshold = NULL ; df3$Threshold = NULL
#retirando as colunas desnecessárias
  return(df3)
}
```

Help

```
idarea      package:unkown         R Documentation

Separação áreas de reprodução, migração e alimentação
Description:
~~ A função "idearea" é utilziada para separação das áreas de reprodução,
migração e alimentação por transmissão satelital pelo sistema ARGOS. Este é
especilamente indicado para estudos em populações de baleias-jubarte do
Hemisfério Sul. ~~

Usage:
~~ idarea(df) ~~
Arguments:
~~ df - data.frame contendo pelo menos as colunas Ptt, Date, Quality,
Latitude e Longitude  ~~

Details:
~~ Dados do sistema ARGOS possibilita a análise de movimentos dos animais.
Eles contém a descrição da data e hora, latiutode e longitude e qualidade de
cada localização por animal. ~~
~~ A função identifica as três áreas (reprodução = AR, migração = AM e
alimentação = AA), podendo ser aplicada em diferentes populações e de acordo
com a área de onde foram implantados os transmissores, na área de reprodução
ou alimentação. ~~

Value:
~~ df - o data.frame do input terá como resultado uma nova coluna com a
especificação da área (AR, AM e AA) identificada de acordo com sua
trajetória para cada animal. ~~

Warning:
~~ Se o transmissor foi implantado na AR, irá calcular a partir desta área e
ignorar o pressuposto da AA. Se o transmissor foi implantado na AA, irá
```

```
calcular a partir desta área e ignorar o pressuposto da AR.  ~~
Author:
~~ Érika Coelho <erika.coelho@ecologia.ufjf.br> - Mestranda pelo Programa de
Pós-graduação de Ecologia na Universidade Federal de Juiz de Fora (UFJF). ~~
References:
~~ WINN & REICHLEY, 1985; KATONA & WHITEHEAD, 1981; International Whaling
Comission, 1998; GARRIGUE et al. 2010; ARGOS, 1990. ~~

Examples:
~~ x <- idarea(df, area = T)  ~~
```