Plot positions only

Plots the GPS positions as eastings and northings on a map.

```
rm(list=ls())
setwd("~/WORKSHOP/GPS/")
library(dplyr)
```

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

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

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

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

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

```
library(MASS)
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
library(dsm)
```

```
## Loading required package: mgcv
```

```
## Loading required package: nlme
```

```
##
## Attaching package: 'nlme'
```

```
## The following object is masked from 'package:dplyr':
##
##     collapse
```

```
## This is mgcv 1.8-39. For overview type 'help("mgcv-package")'.
```

```
## Loading required package: mrds
```

```
## This is mrds 2.2.6
## Built: R 3.6.3; ; 2022-06-17 10:42:29 UTC; unix
```

```
## Loading required package: numDeriv
```

```
## This is dsm 2.3.2
## Built: R 3.6.3; ; 2022-06-17 10:42:38 UTC; unix
```

```
library(anytime)

Rearth <- 6371*1e3 # meters

data <- readRDS("OUTPUT/complete_GPS_data.rds")
data$Longitude <- as.numeric(data$Longitude)
data$Latitude <- as.numeric(data$Latitude)
data <- unique(data)
IDs <- sort(unique(data$UnitId))
# cull on date
idx <- which(month(data$POSIX) > 3)
data <- data[idx,]
# get the list of Station numbers vs station names
namesnumbers <- read.csv("DATA/names_numbers.txt",sep="",header=F)
namesnumbers[,1] <- as.character(namesnumbers[,1])
```

Utility GC formula

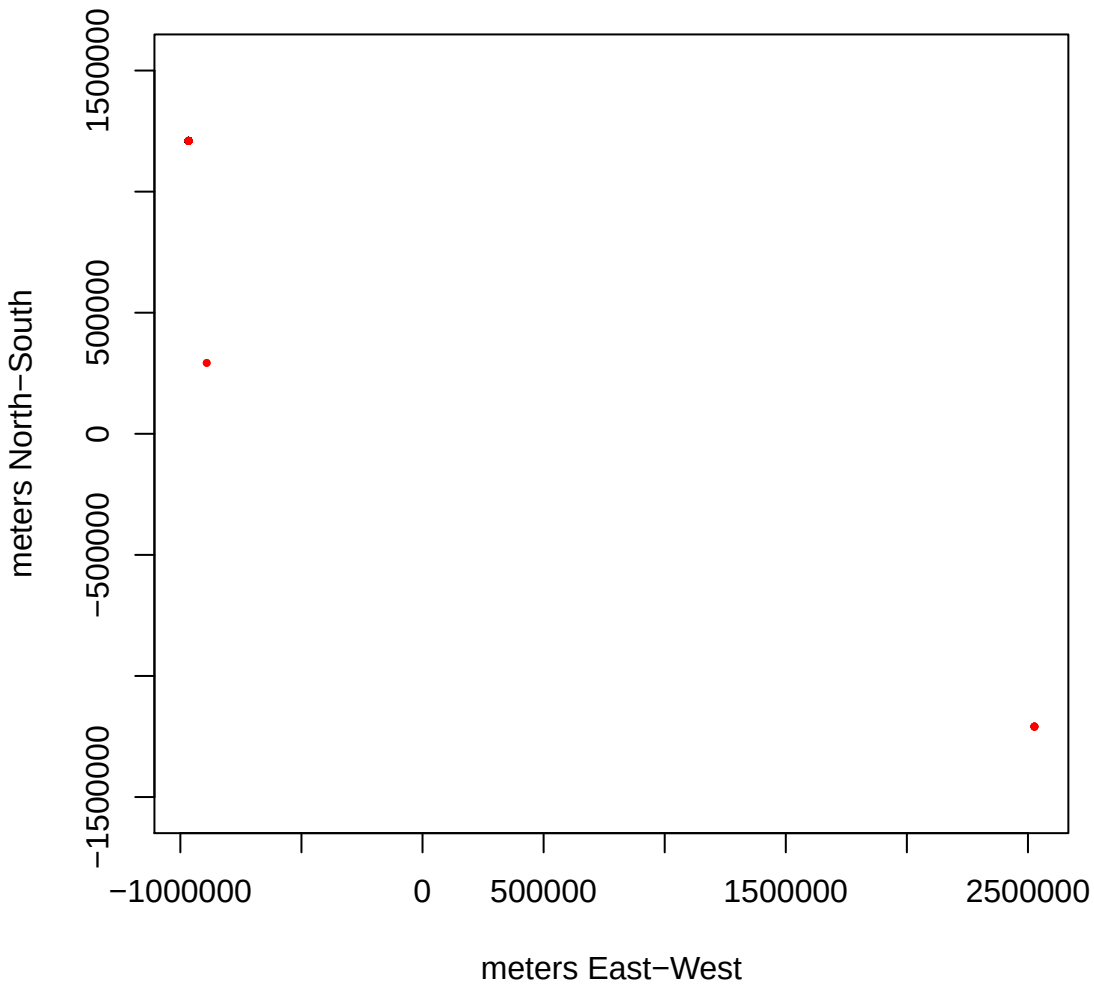
```
# Calculates the geodesic distance between two points specified by radian lat/lon using the
# Haversine formula (hf)
gcd.hf <- function(long1, lat1, long2, lat2) {
  R <- 6371*1000 # Earth mean radius [m]
  delta.long <- (long2 - long1)
  delta.lat <- (lat2 - lat1)
  a <- sin(delta.lat/2)^2 + cos(lat1) * cos(lat2) * sin(delta.long/2)^2
  c <- 2 * asin(min(1,sqrt(a)))
  d = R * c
  return(d) # Distance in m
}
```

read and plot each file

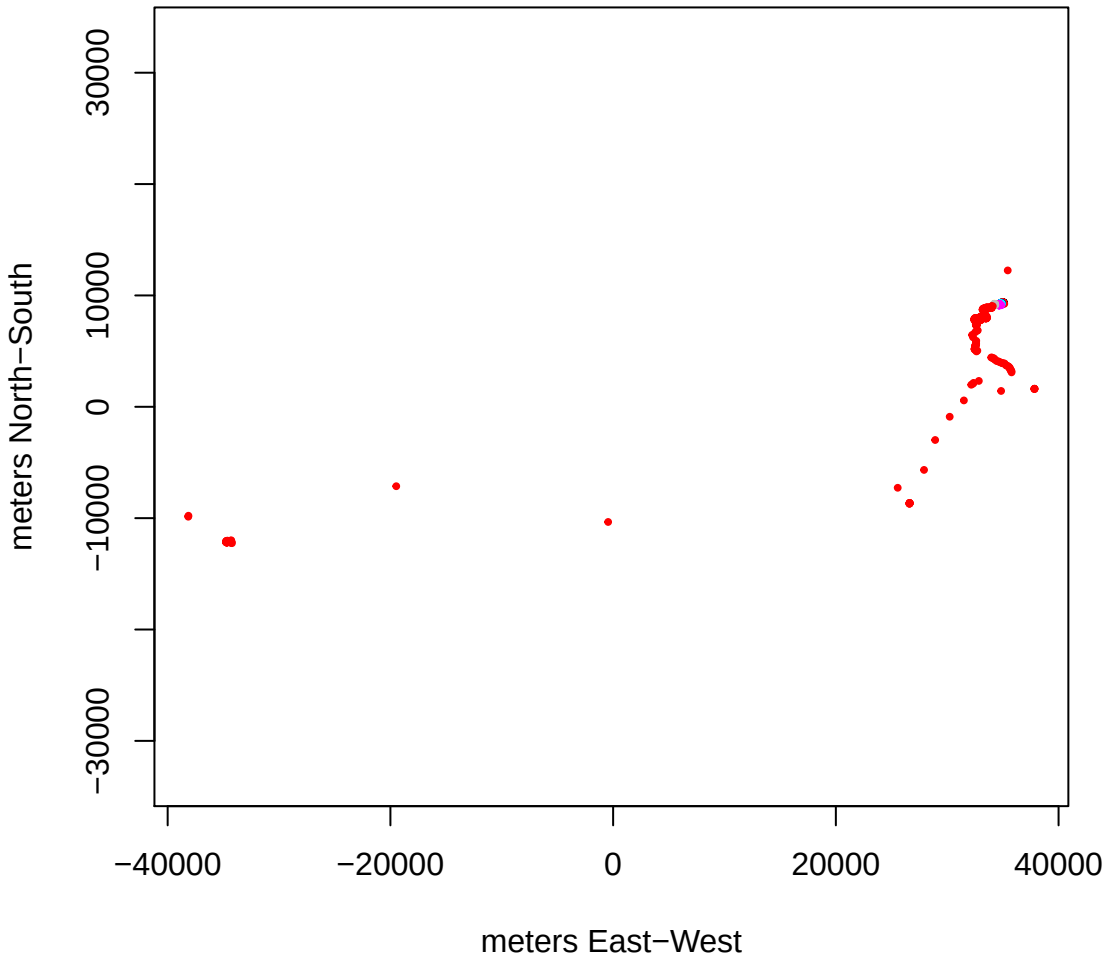
```
# list the important times - start, jumps, ending:
important_times <- c(as.POSIXct("2022-03-31 00:00:00",tz="UTC"),as.POSIXct("2022-04-07 00:00:00",tz="UTC
                     as.POSIXct("2022-04-07 00:00:00",tz="UTC"),as.POSIXct("2022-04-24 12:00:00",tz="UTC
                     as.POSIXct("2022-04-24 12:00:00",tz="UTC"),as.POSIXct("2022-04-27 12:00:01",tz="UTC
                     as.POSIXct("2022-04-27 12:00:01",tz="UTC"),as.POSIXct("2022-05-03 03:00:00",tz="UTC
                     as.POSIXct("2022-05-03 03:00:00",tz="UTC"),as.POSIXct("2022-05-19 22:00:00",tz="UTC
                     as.POSIXct("2022-05-19 22:00:00",tz="UTC"),as.POSIXct("2022-06-09 02:00:00",tz="UTC
                     as.POSIXct("2022-06-09 02:00:00",tz="UTC"),as.POSIXct("2022-06-15 02:00:00",tz="UTC
                     as.POSIXct("2022-06-15 02:00:00",tz="UTC"),as.POSIXct("2022-06-25 00:00:00",tz="UTC
                     as.POSIXct("2022-06-25 00:00:00",tz="UTC"),as.POSIXct("2022-06-30 23:00:00",tz="UTC
                     as.POSIXct("2022-06-30 23:00:00",tz="UTC"),as.POSIXct("2023-06-20 02:00:00",tz="UTC
list_of_eventful_dates <- unique(sort(important_times))
limitdates <- NULL
for (it in seq(from=1,to=length(important_times),by=2))
```

```
{  limitdates <- rbind(limitdates,c(anytime(important_times[it],asUTC=T),anytime(important_times[it+1],a


for (ifil in IDs)
{
  name <- ifil
  print("-----------------------------------------")
  print(paste(" Processing file ",name))

  sdx <- which(data$UnitId == ifil)
  df <- data[sdx,]
  df <- na.omit(df)
  xy <- latlong2km(df$Longitude, df$Latitude)

  #plot(xy$km.e*1000,xy$km.n*1000,asp=1,main=namesnumbers[which(namesnumbers[,2] == name),1],type="p",p
  plot(xy$km.e*1000,xy$km.n*1000,asp=1,main=namesnumbers[which(namesnumbers[,2] == name),1],type="p",pch
  # overplot colours for each segment
  for (iseg in 1:nrow(limitdates))
  {
    idx <- which(df$POSIX >= limitdates[iseg,1] & df$POSIX < limitdates[iseg,2])
    points(xy$km.e[idx]*1000,xy$km.n[idx]*1000,col=iseg,pch=19,cex=0.4)
  } # end iseg loop
  print("-----------------------------------------")
} # end ifil loop
```

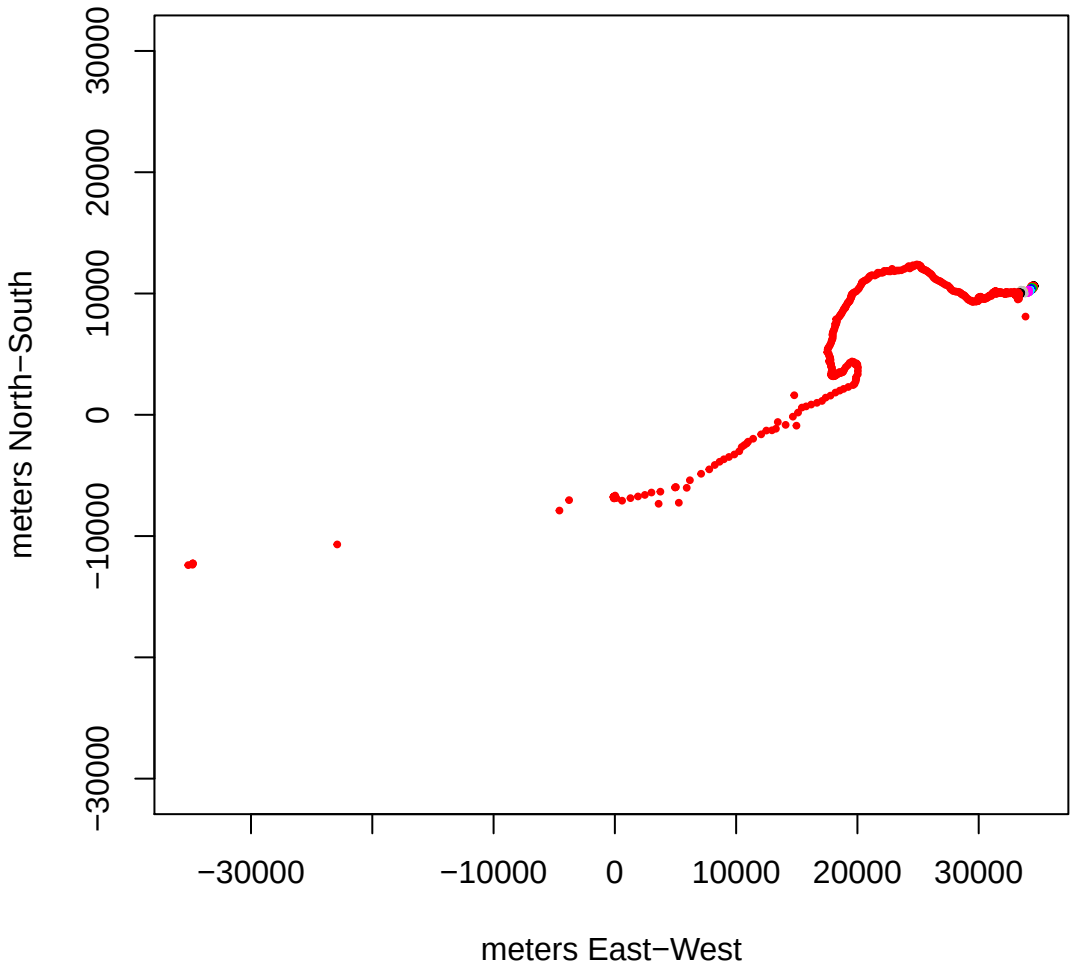
```
## [1] "-----------------------------------------"
## [1] " Processing file  81619"
```



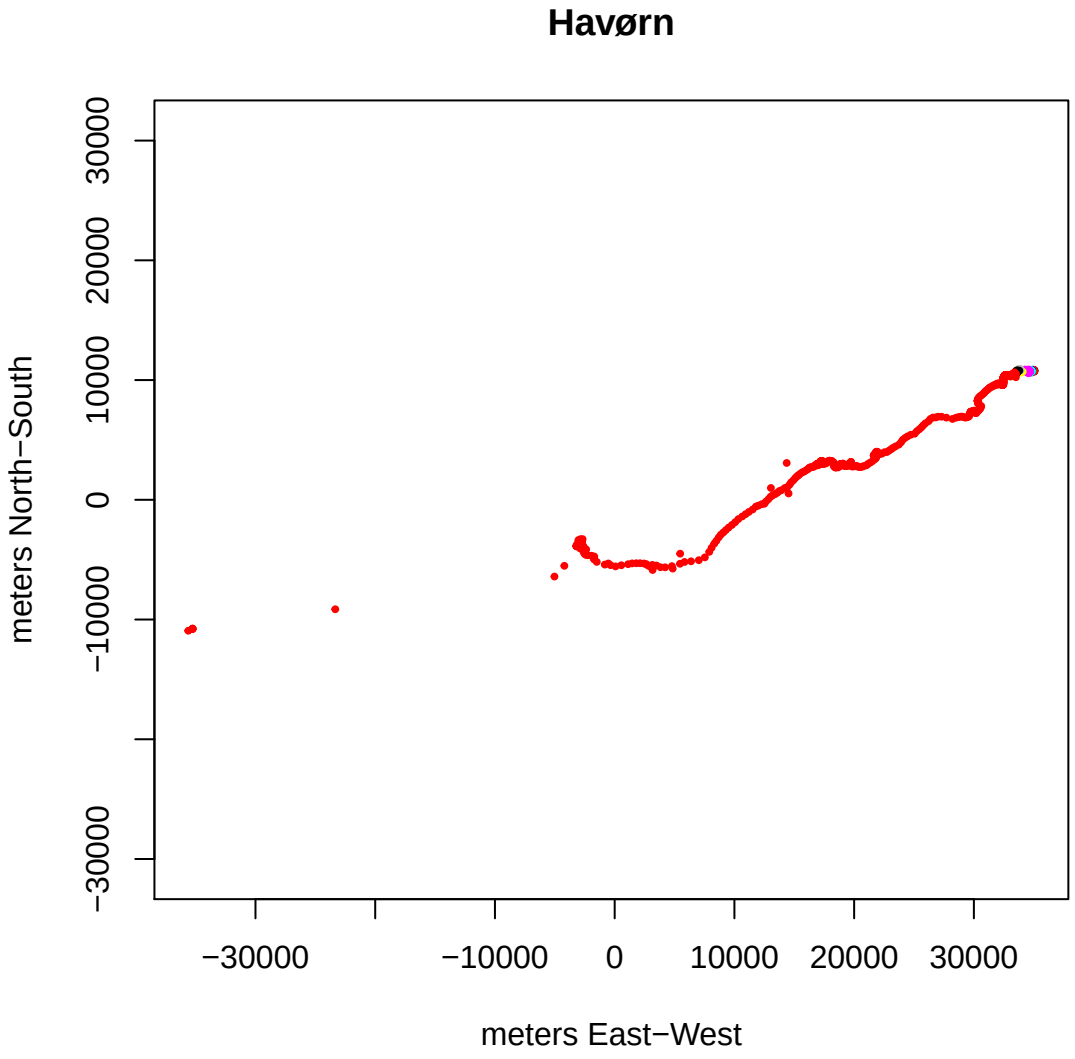

```
## [1] "----------------------------------------"
## [1] "----------------------------------------"
## [1] " Processing file  88319"
```



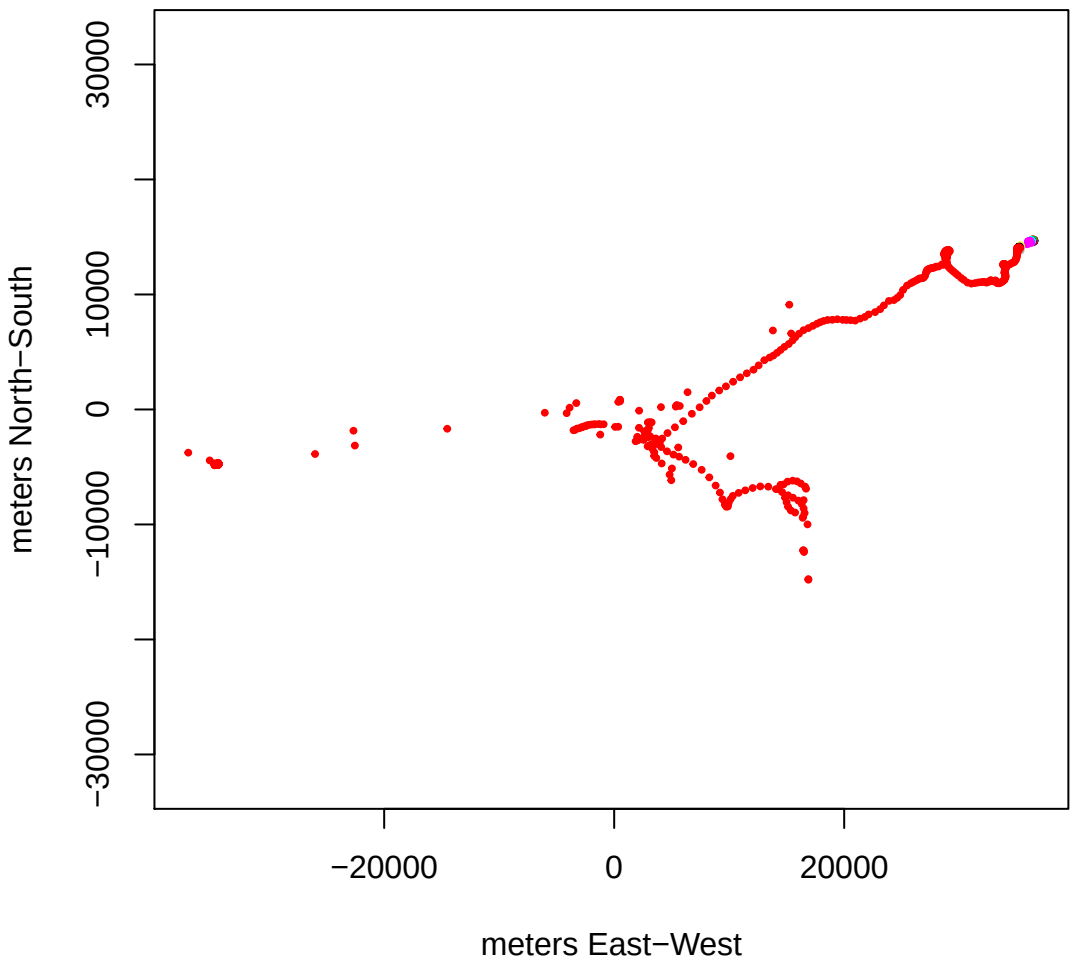

```
## [1] "----------------------------------------"
## [1] "----------------------------------------"
## [1] " Processing file  88462"
```



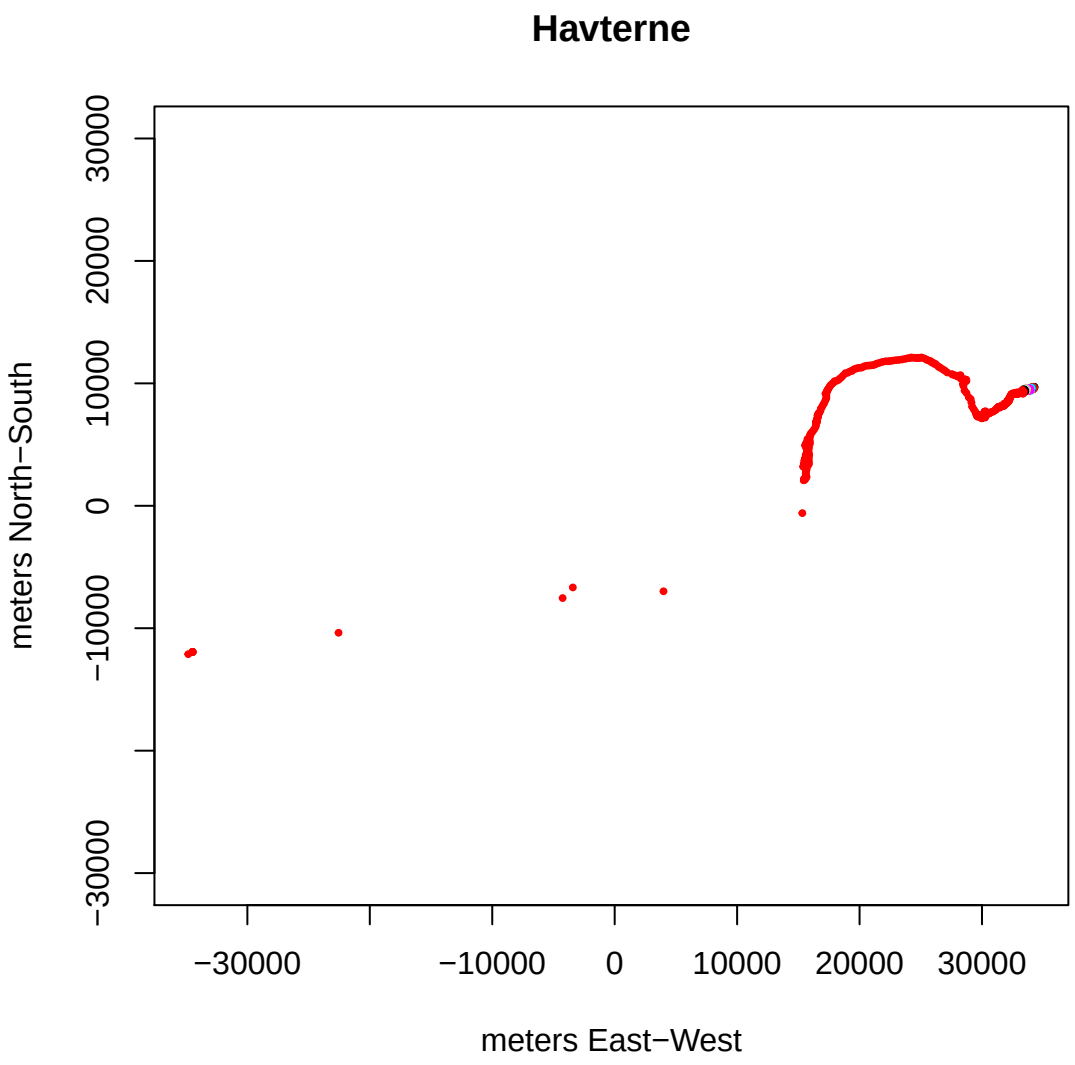

```
## [1] "----------------------------------------"
## [1] "----------------------------------------"
## [1] " Processing file  88463"
```



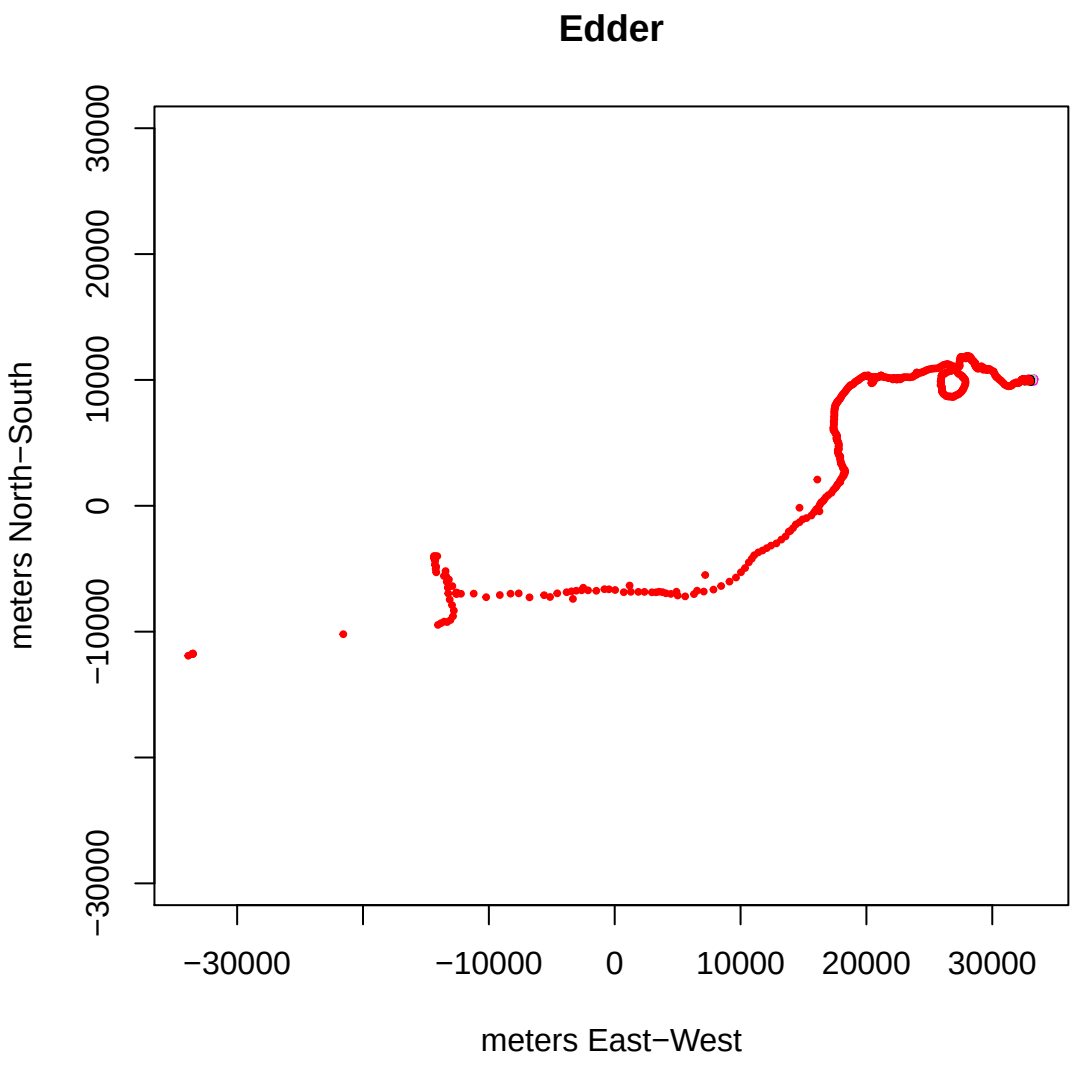

```
## [1] "----------------------------------------"
## [1] "----------------------------------------"
## [1] " Processing file  88464"
```




```
## [1] "----------------------------------------"
## [1] "----------------------------------------"
## [1] " Processing file  88465"
```




```
## [1] "----------------------------------------"
## [1] "----------------------------------------"
## [1] " Processing file  88617"
```




```
## [1] "----------------------------------------"
```

Plot coloured points

```
plotcolouredpoints <- function(x,y,limitdates,ipair,ivar)
{
  idx <- which(df$UTC >= limitdates[ipair,1] & df$UTC < limitdates[ipair,2])
  points(x[idx],y[idx],type="p",cex=0.3,col=1+ipair)
}
```

Model motion 2

model positions and calculate speeds at jumps

```
model_motion2 <- function(df,name,limitdates)
{

  par(mfrow=c(2,1))
  nlimits <- nrow(limitdates)
  statname <- name #strsplit(strsplit(name, "/")[[1]][2],".rds")[[1]][1]
  # Latitude
  latitude_pred_at_interval_left_right <- NULL
  lat_speed <- NULL
  # loop over limidates and model positions before and after each limitdate
  #plot(df$POSIX,df$Latitude,type="p",xlim=range(df$POSIX),ylim=range(df$Latitude,na.rm=T),xlab="Time",
  plot(df$POSIX,df$Latitude,type="p",xlim=range(df$POSIX),ylim=range(df$Latitude,na.rm=T),xlab="Time",yl
  for (ilimit in 1:nlimits)
  {
    idx <- which(df$POSIX >= limitdates[ilimit,1] & df$POSIX < limitdates[ilimit,2] & !is.na(df$Latitude
    if (length(idx) != 0){
      rlmfit <- rlm(df$Latitude[idx] ~ df$POSIX[idx])
      lat_speed <- rbind.data.frame(lat_speed,c(ilimit,summary(rlmfit)$coefficients[2]*3600/180*pi)) #
      if (ilimit == 1) {
        #plot(df$POSIX[idx],df$Latitude[idx],type="p",xlim=range(df$POSIX),ylim=range(df$Latitude,na.rm
        points(df$POSIX[idx],df$Latitude[idx])
        lines(df$POSIX[idx],rlmfit$fitted.values,col=2,lwd=3)
        # evaluate diff at jump
        latitude_pred_at_interval_left_right  <- c(first(rlmfit$fitted.values), last(rlmfit$fitted.value
      }
      if (ilimit  > 1) {
        points(df$POSIX[idx],df$Latitude[idx])
        lines(df$POSIX[idx],rlmfit$fitted.values,col=2,lwd=3)
        #
        latitude_pred_at_interval_left_right  <- rbind.data.frame(latitude_pred_at_interval_left_right,
      }
    } # end length(idx)
  }
  colnames(lat_speed) <- c("segment_number","lat_speed_radperhr")
  # Longitude
  longitude_pred_at_interval_left_right <- NULL
  lon_speed <- NULL
  # loop over limidates and model positions before and after each limitdate
  #plot(df$POSIX,df$Longitude,type="p",xlim=range(df$POSIX),ylim=range(df$Longitude,na.rm=T),xlab="Time
  plot(df$POSIX,df$Longitude,type="p",xlim=range(df$POSIX),ylim=range(df$Longitude,na.rm=T),xlab="Time"
  for (ilimit in 1:nlimits)
  {
    idx <- which(df$POSIX >= limitdates[ilimit,1] & df$POSIX < limitdates[ilimit,2] & !is.na(df$Longitud
    if (length(idx) != 0){
      #print(c("f",length(idx)))
      rlmfit <- rlm(df$Longitude[idx] ~ df$POSIX[idx])
      lon_speed <- rbind.data.frame(lon_speed,c(ilimit,summary(rlmfit)$coefficients[2]*3600/180*pi)) #
      #print(c(rlmfit$fitted.values[1],last(rlmfit$fitted.values)))
      if (ilimit == 1) {
        #browser()
        #plot(df$POSIX[idx],df$Longitude[idx],type="p",xlim=range(df$POSIX),ylim=range(df$Longitude,na.
        points(df$POSIX[idx],df$Longitude[idx])
        lines(df$POSIX[idx],rlmfit$fitted.values,col=2,lwd=3)
```

```
        # evaluate diff at jump
        longitude_pred_at_interval_left_right  <- c(first(rlmfit$fitted.values), last(rlmfit$fitted.valu
      }
      if (ilimit  > 1) {
        points(df$POSIX[idx],df$Longitude[idx])
        lines(df$POSIX[idx],rlmfit$fitted.values,col=2,lwd=3)
        #
        longitude_pred_at_interval_left_right  <- rbind.data.frame(longitude_pred_at_interval_left_righ

      }
    }
  }
  colnames(lon_speed) <- c("segment_number","lon_speed_radperhr")

  segment_speed <- Rearth*sqrt((lon_speed[,2]*cos(median(df$Latitude,na.rm=T)/180*pi))^2+(lat_speed[,2]

  return(list("lats"=latitude_pred_at_interval_left_right,"longs"=longitude_pred_at_interval_left_right
}
```

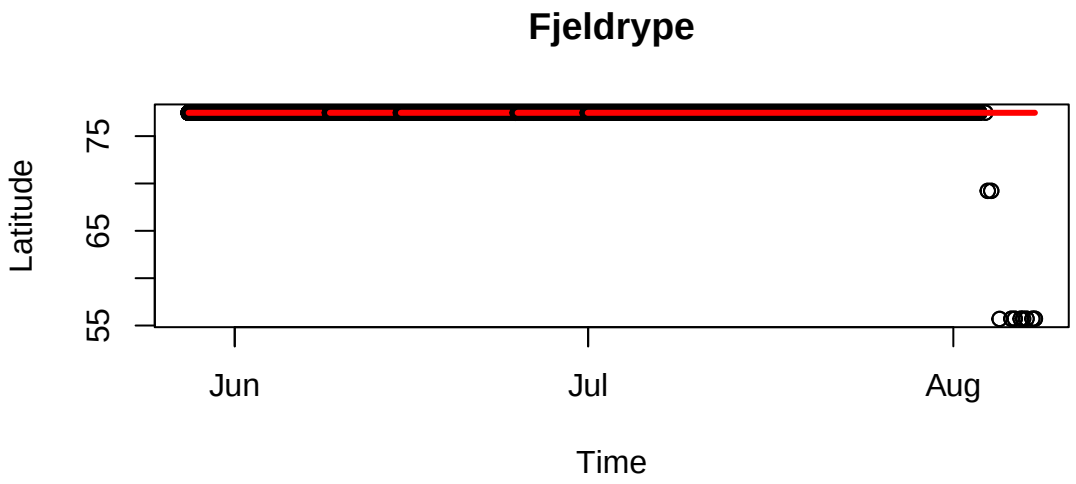
function to get interval and jump speeds

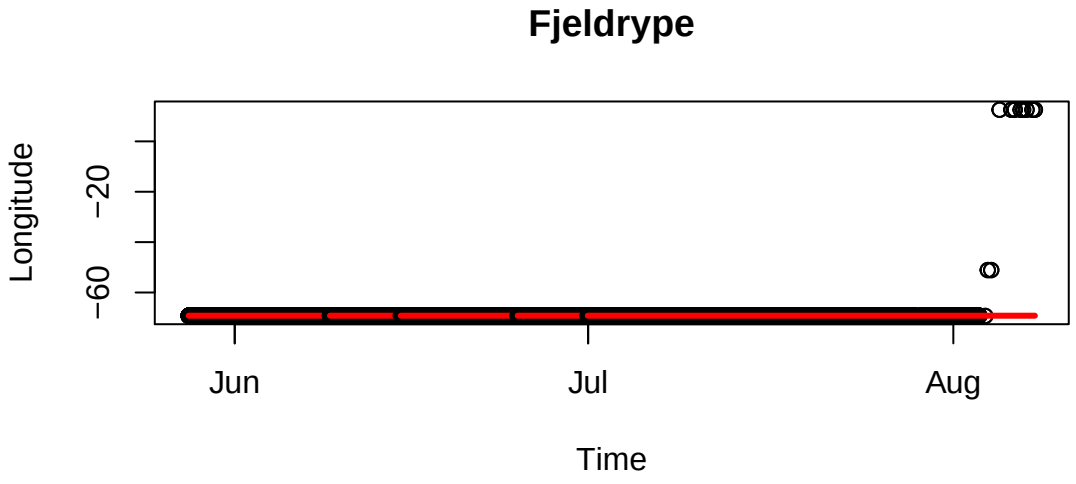
```
get_surge_speeds <- function(listerne,lon_in,lat_in)
{

  lon <- lon_in/180*pi # in radians
  lat <- lat_in/180*pi
  delta_t <- 1 # hours

  lon_here <- listerne$longs
  lat_here <- listerne$lats

  # calculate jump speeds
  n_segments <- nrow(listerne$lats)
  speed <- NULL
  for (iseg in 1:(n_segments-1))
  {
    delta_lon <- (lon_here[iseg,2]-lon_here[iseg+1,1])/180*pi
    delta_lat <- (lat_here[iseg,2]-lat_here[iseg+1,1])/180*pi

    speed <- rbind.data.frame(speed,c(iseg,Rearth/delta_t*sqrt(delta_lon^2*cos(lat)^2+delta_lat^2))) #
  }
  colnames(speed) <- c("jump_number","speed_metersph")
  #browser()
  return(list("speed_jump"=speed))
}
```

read and plot each file

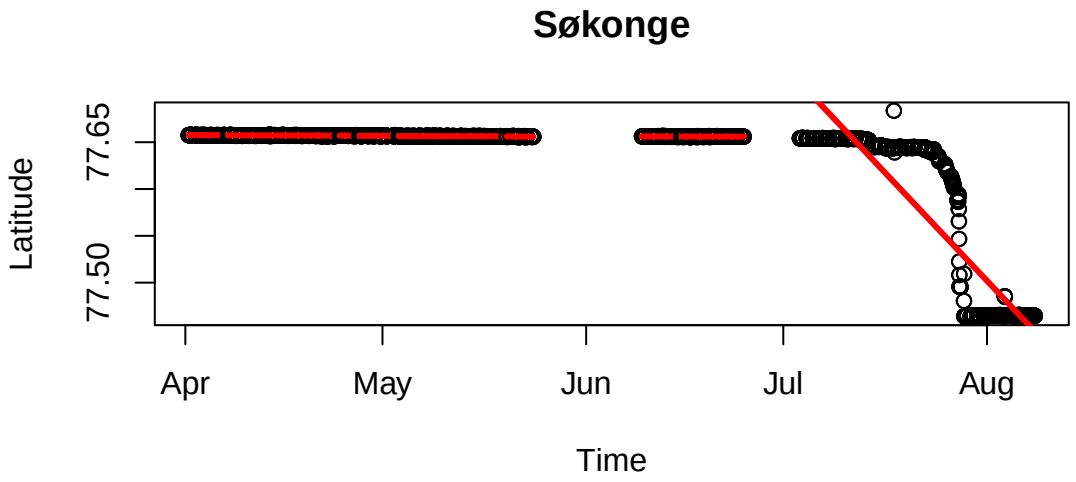
```
for (id in IDs)

{
  print("-----------------------------------------")
  print(paste(" Processing unitID ",id))
  mdx <- which(data$UnitId == id)
  df <- data[mdx,]

  # model speed as unconnected straight line segments
  listerne <- model_motion2(df,id,limitdates)

  segspeeds <- round(listerne$segspeed,2)
  print("Segment speeds in m/hr : ")
  print(segspeeds)

  # get surge speeds
  speeds <- get_surge_speeds(listerne,lon=median(df$Longitude,na.rm=T),lat=median(df$Latitude,na.rm=T))
  print(speeds)
  #browser()
}
```

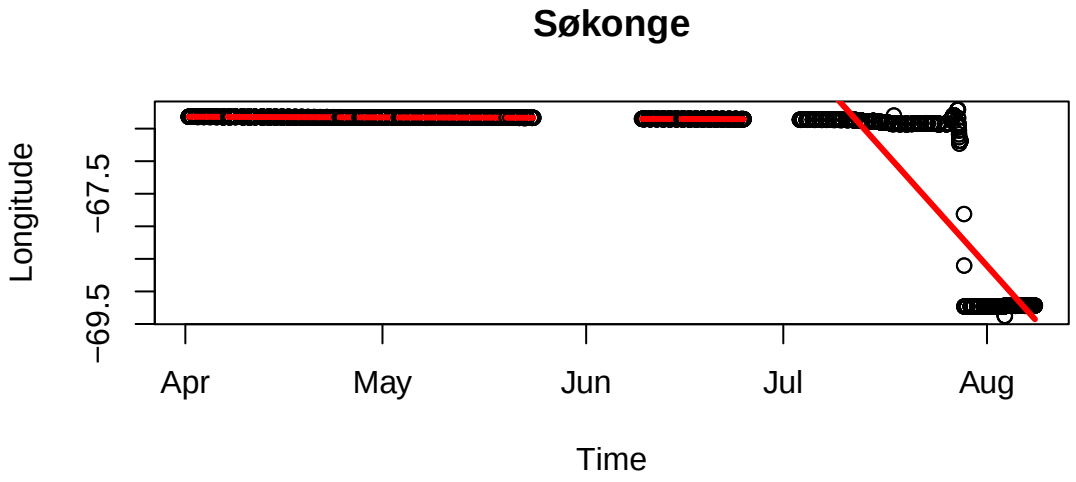
```
## [1] "-----------------------------------------"
## [1] " Processing unitID  81619"
```





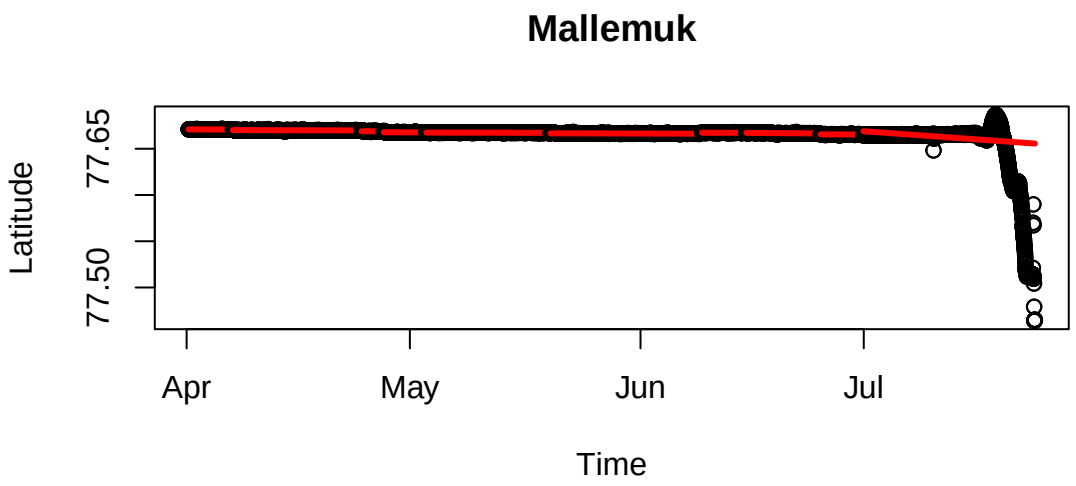

```
## [1] "Segment speeds in m/hr : "
## [1] 0.01 0.02 0.00 0.05 0.01
## $speed_jump
##   jump_number speed_metersph
## 1           1       4.097393
## 2           2       1.781471
## 3           3       4.962987
## 4           4       8.292684
##
## [1] "----------------------------------------"
## [1] " Processing unitID  88319"
```

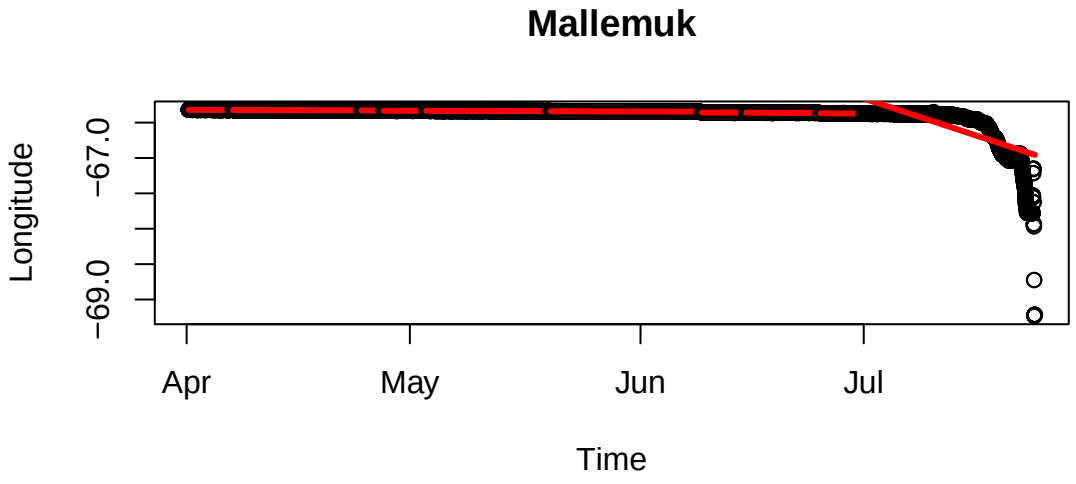




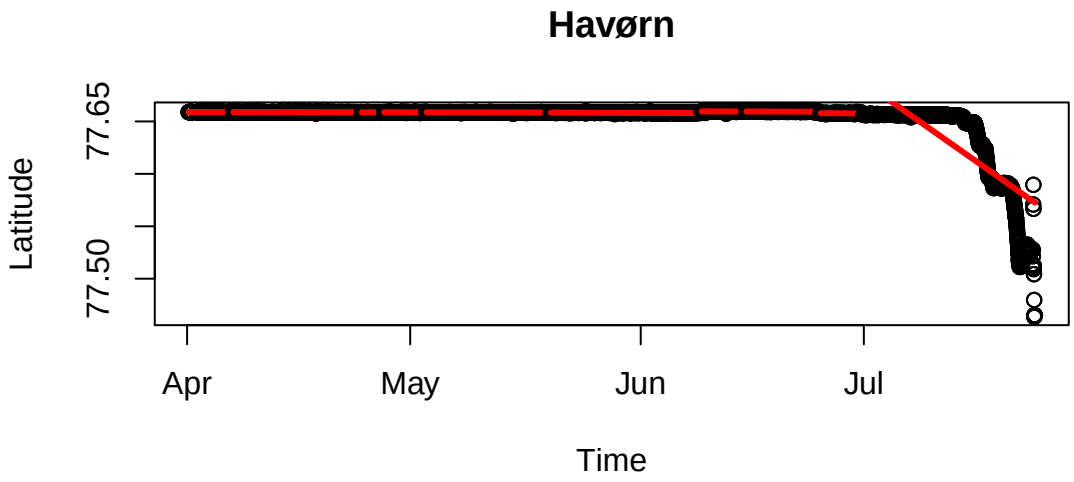

```
## [1] "Segment speeds in m/hr : "
## [1]   0.32   0.32   0.39   0.29   0.23   0.49   0.16   0.27 116.63
## $speed_jump
##   jump_number speed_metersph
## 1           1      173.92487
## 2           2      148.86949
## 3           3       69.32871
## 4           4      147.60891
## 5           5      154.55567
## 6           6      451.08760
## 7           7       88.65181
## 8           8    76669.56621
##
## [1] "----------------------------------------"
## [1] " Processing unitID  88462"
```

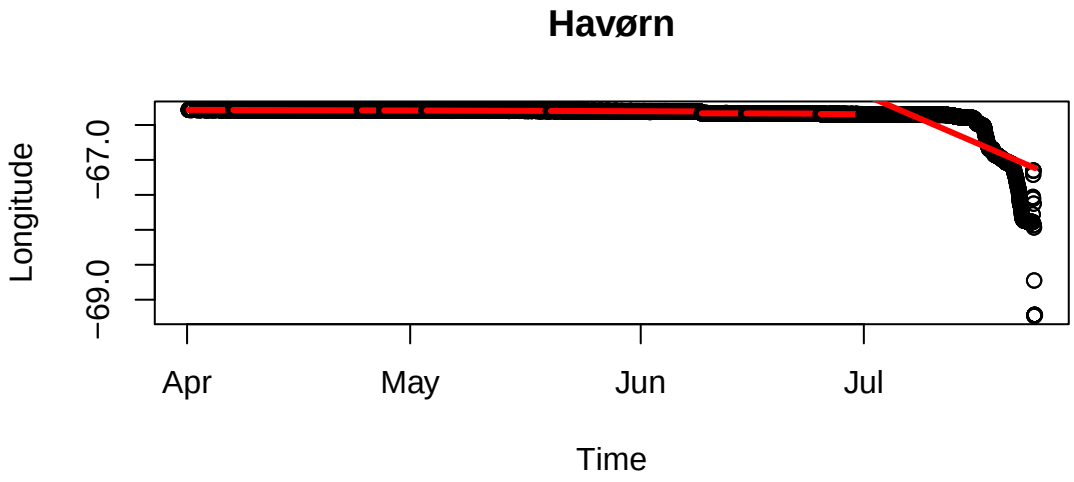
```
## Warning in rlm.default(x, y, weights, method = method, wt.method = wt.method, :
## 'rlm' failed to converge in 20 steps
```



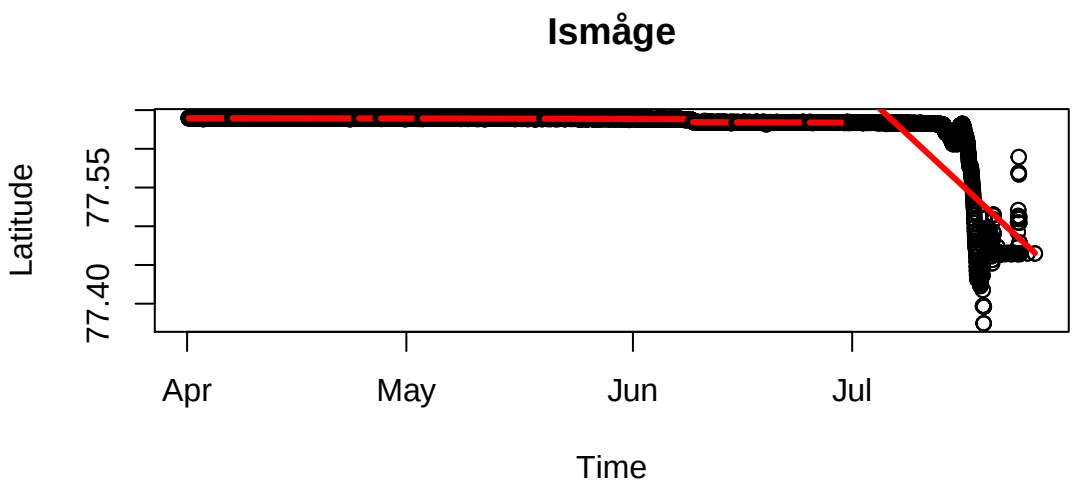


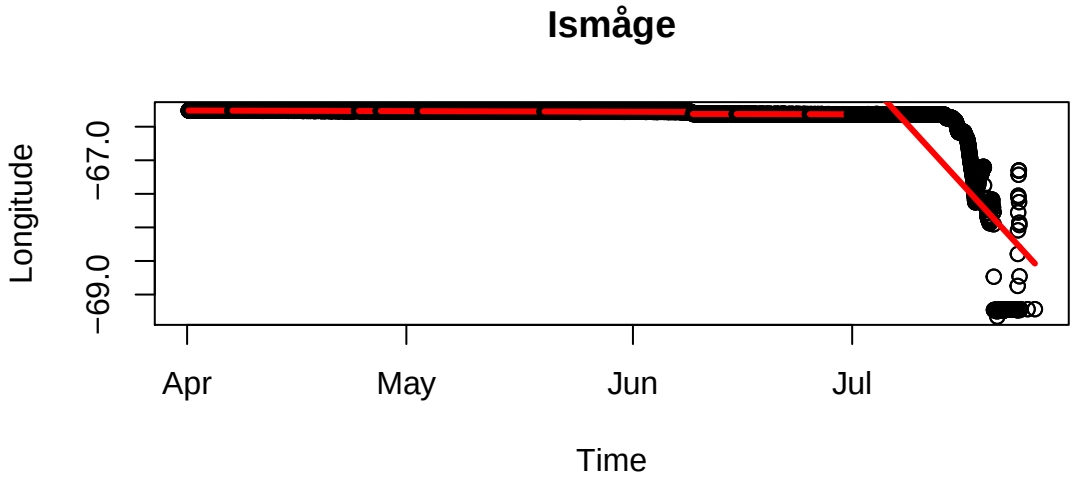

```
## [1] "Segment speeds in m/hr : "
##  [1]  0.33  0.26  0.51  0.23  0.25  0.28  0.18  0.35  0.55 34.35
## $speed_jump
##   jump_number speed_metersph
## 1           1       214.5512
## 2           2       215.5838
## 3           3       197.1383
## 4           4       151.5050
## 5           5       289.2911
## 6           6       490.6288
## 7           7       111.3171
## 8           8       347.3581
## 9           9     13986.8623
##
## [1] "-----------------------------------------"
## [1] " Processing unitID  88463"
```





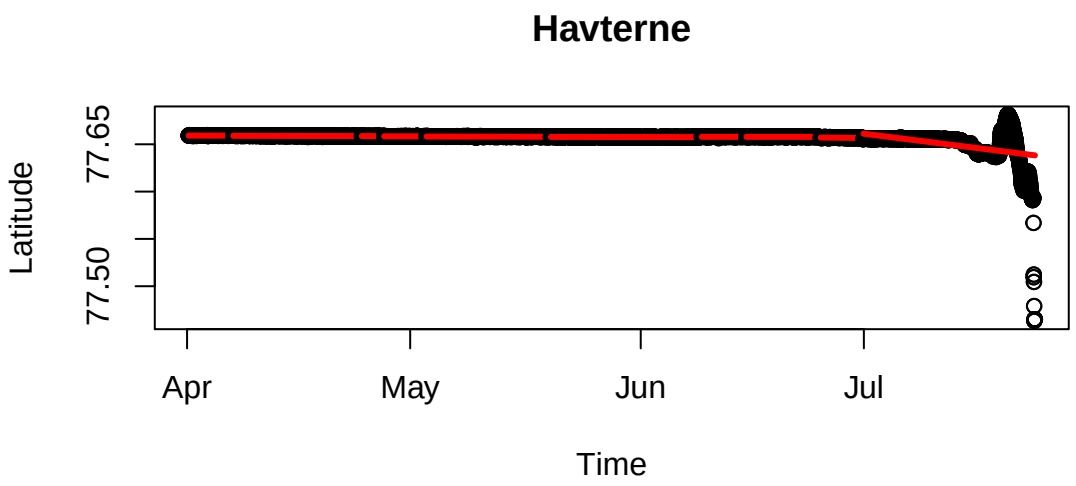

```
## [1] "Segment speeds in m/hr : "
##  [1]  0.30  0.31  0.32  0.29  0.33  0.37  0.21  0.33  0.53 50.90
## $speed_jump
##   jump_number speed_metersph
## 1           1      169.33510
## 2           2      150.41577
## 3           3       64.25379
## 4           4      170.98595
## 5           5      309.75666
## 6           6      834.85394
## 7           7      107.20882
## 8           8      353.45754
## 9           9    20696.28619
##
## [1] "-----------------------------------------"
## [1] " Processing unitID  88464"
```

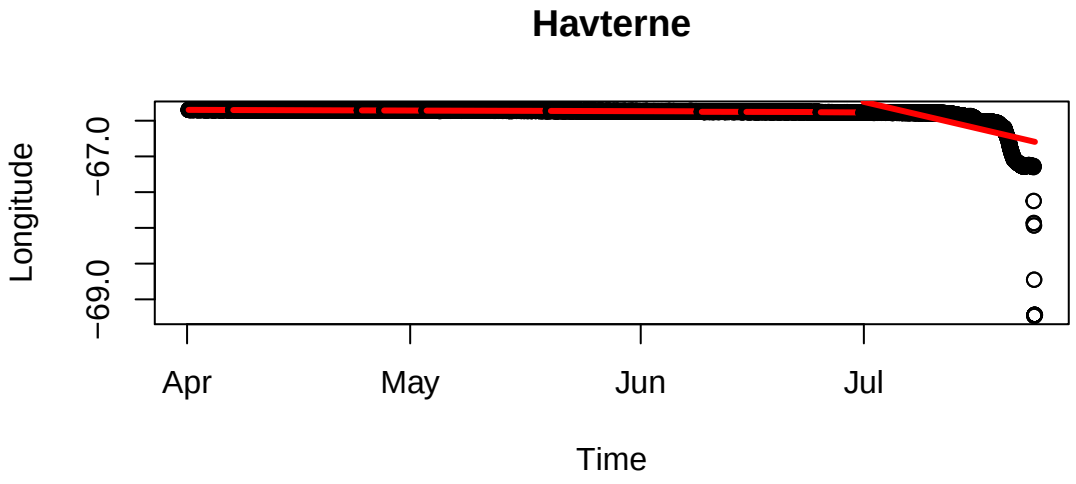




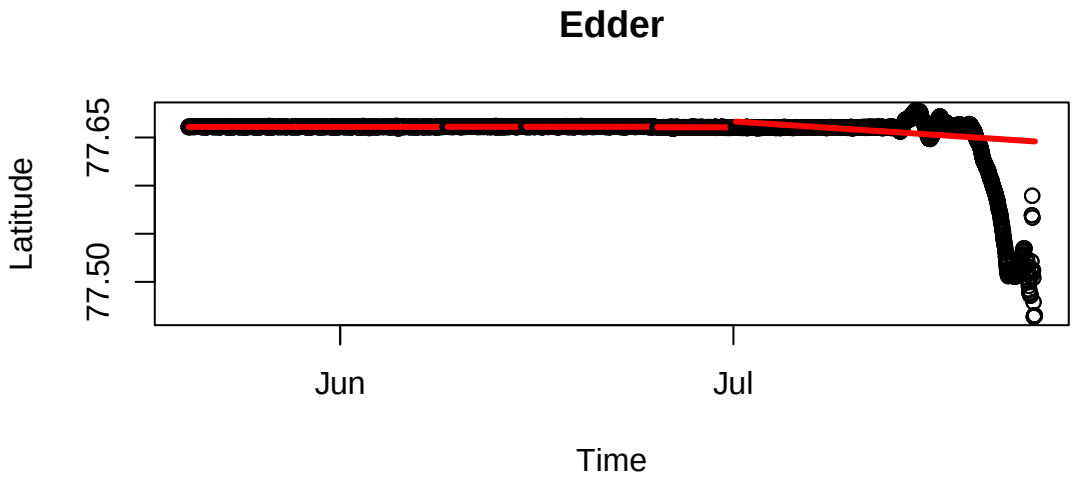

```
## [1] "Segment speeds in m/hr : "
##  [1]   0.28   0.29   0.28   0.31   0.31   0.37   0.14   0.23   0.53 124.64
## $speed_jump
##   jump_number speed_metersph
## 1           1      161.74517
## 2           2      142.69312
## 3           3       62.05809
## 4           4      168.90758
## 5           5      299.03470
## 6           6     1029.01771
## 7           7       77.31955
## 8           8      123.99654
## 9           9    56184.47986
##
## [1] "-----------------------------------------"
## [1] " Processing unitID  88465"
```

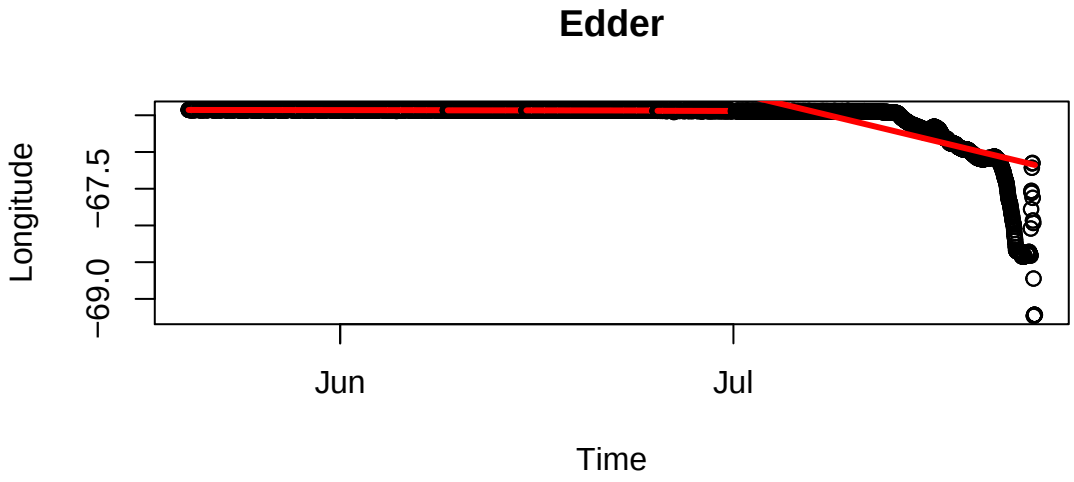
```
## Warning in rlm.default(x, y, weights, method = method, wt.method = wt.method, :
## 'rlm' failed to converge in 20 steps
```






```
## [1] "Segment speeds in m/hr : "
##  [1]  0.28  0.31  0.37  0.26  0.25  0.27  0.15  0.20  0.61 24.53
## $speed_jump
##   jump_number speed_metersph
## 1           1      187.53561
## 2           2      151.99784
## 3           3       65.96263
## 4           4      150.58500
## 5           5      245.31215
## 6           6      314.01230
## 7           7       70.37069
## 8           8      225.83068
## 9           9    10073.11763
##
## [1] "-----------------------------------------"
## [1] " Processing unitID  88617"
## Warning in rlm.default(x, y, weights, method = method, wt.method = wt.method, :
## 'rlm' failed to converge in 20 steps
```






```
## [1] "Segment speeds in m/hr : "
## [1]  0.15  0.11  0.20  0.35 42.75
## $speed_jump
##   jump_number speed_metersph
## 1           1      132.88178
## 2           2       55.49541
## 3           3      202.41097
## 4           4    17594.41999
```

```
print("-----------------------------------------")
```

```
## [1] "-----------------------------------------"
```

Summary of eventful dates

```
list_of_eventful_dates
```

```
##  [1] "2022-03-31 00:00:00 UTC" "2022-04-07 00:00:00 UTC"
##  [3] "2022-04-24 12:00:00 UTC" "2022-04-27 12:00:01 UTC"
##  [5] "2022-05-03 03:00:00 UTC" "2022-05-19 22:00:00 UTC"
```

```
##  [7] "2022-06-09 02:00:00 UTC" "2022-06-15 02:00:00 UTC"
##  [9] "2022-06-25 00:00:00 UTC" "2022-06-30 23:00:00 UTC"
## [11] "2023-06-20 02:00:00 UTC"
```