Plot positions and speeds

Version 5. Like v 4, buta nnotaes the plots with speeds of the smooth motion as well as the surges. Version 4: plots from GPS data previously fetched from server with code "xxx.Rmd"

```
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(anytime)
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
```

```
Rearth <- 6371*1e3 # meters

delta_t <- 0.66 # hours - estimated duration of a 'surge' (i.e. 1 time step in the data)
```

```r
# read data stored previously by code "GPS_from_server_1.Rmd"
data <- readRDS("OUTPUT/updated_file_of_GPS_observations.rds")
#data$POSIX <- as.POSIXct(data$UTC,tz="UTC")
idx <- which(data$Month < 4)
data <- data[-idx,] # skip all data before March 31st
unitIDs <- sort(unique(data$UnitID))
#idx <- order(unitIDs)
#unitIDs <- unitIDs[rev(idx)]
#=============================================================================
# Have to patch Mallemuk from older file
Mallemuk <- readRDS("OUTPUT/Mallemuk.rds")
Mallemuk$`Timestamp UTC` <- as.POSIXct(Mallemuk$`Timestamp UTC`,tz="UTC")
nyM <- Mallemuk[,c("Longitude","Latitude","Timestamp UTC")]
nyM$Year <- year(nyM$`Timestamp UTC`)
nyM$Month <- month(nyM$`Timestamp UTC`)
nyM$Day <- day(nyM$`Timestamp UTC`)
nyM$Hour <- hour(nyM$`Timestamp UTC`)
nyM$Minute <- minute(nyM$`Timestamp UTC`)
nyM$UnitID <- 88462
nyM <- na.omit(nyM)
nams <- colnames(data)
colnames(nyM) <- nams
# join
data <- rbind.data.frame(data,nyM)
#=============================================================================
# Have to patch Soekonge from older file
Soekonge <- readRDS("OUTPUT/Soekonge.rds")
Soekonge$`Timestamp UTC` <- as.POSIXct(Soekonge$`Timestamp UTC`,tz="UTC")
nyS <- Soekonge[,c("Longitude","Latitude","Timestamp UTC")]
nyS$Year <- year(nyS$`Timestamp UTC`)
nyS$Month <- month(nyS$`Timestamp UTC`)
nyS$Day <- day(nyS$`Timestamp UTC`)
nyS$Hour <- hour(nyS$`Timestamp UTC`)
nyS$Minute <- minute(nyS$`Timestamp UTC`)
nyS$UnitID <- 88319
nyS <- na.omit(nyS)
nams <- colnames(data)
colnames(nyS) <- nams
# join
data <- rbind.data.frame(data,nyS)
# remove duplicated rows across dataframe
hej <- data %>%  distinct(.keep_all = FALSE)
data <- hej
write.table(data,"OUTPUT/dummy.csv",sep=";",quote = F,row.names = F)
data <- read.csv2("OUTPUT/dummy.csv",sep=";",stringsAsFactors = FALSE)
data$lon <- as.numeric(data$lon)
data$lat <- as.numeric(data$lat)
data$POSIX <- ISOdatetime(data$Year,data$Month,data$Day,data$Hour,data$Minute,sec=0,tz="UTC")
unitIDs <- sort(unique(data$UnitID))
saveRDS(data,"OUTPUT/combined_GPS_data.rds")
```

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
}
```

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)
}
```

function to plot positions and speeds etc

```
plot_stuff <- function(df,name,limitdates)
{
  #browser()
  par(mfrow=c(3,1))
  npairs <- nrow(limitdates)
  statname <- name
  # First plot positions
  plot(df$lon,df$lat,type="p",cex=0.3,xlab="lon",ylab="lat",main=statname)
  for (ipair in 1:npairs){  plotcolouredpoints(df$lon,df$lat,limitdates,ipair,'')  }
  # Plot lon vs time
  plot(df$POSIX,df$lon,type="p",cex=0.3,xlab="Date/Time",ylab="lon",main=statname)
  for (ipair in 1:npairs){  plotcolouredpoints(df$POSIX,df$lon,limitdates,ipair,'')  }
  # Plot lat vs time
  plot(df$POSIX,df$lat,type="p",cex=0.3,xlab="Date/Time",ylab="lat",main=statname)
  for (ipair in 1:npairs){  plotcolouredpoints(df$POSIX,df$lat,limitdates,ipair,'')  }
  print('Returning from plot_stuff')
}
```

Model motion 2

model positions and calculate speeds at jumps

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

  par(mfrow=c(3,1))
  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
  for (ilimit in 1:nlimits)
  {
    idx <- which(df$POSIX >= limitdates[ilimit,1] & df$POSIX < limitdates[ilimit,2] & !is.na(df$lat) )
    if (length(idx) != 0){
    print(c("e",length(idx)))
    rlmfit <- rlm(df$lat[idx] ~ df$POSIX[idx])
    lat_speed <- rbind.data.frame(lat_speed,c(ilimit,summary(rlmfit)$coefficients[2]*3600/180*pi)) # ra
    #print(c(rlmfit$fitted.values[1],last(rlmfit$fitted.values)))
    if (ilimit == 1) {
      #browser()
      plot(df$POSIX[idx],df$lat[idx],type="p",xlim=range(df$POSIX),ylim=range(df$lat,na.rm=T),xlab="Tim
      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.values
    }
    if (ilimit  > 1) {
      points(df$POSIX[idx],df$lat[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, c

    }
    }
  }
  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
  for (ilimit in 1:nlimits)
  {
    idx <- which(df$POSIX >= limitdates[ilimit,1] & df$POSIX < limitdates[ilimit,2] & !is.na(df$lon) )
    if (length(idx) != 0){
      print(c("f",length(idx)))
    rlmfit <- rlm(df$lon[idx] ~ df$POSIX[idx])
    lon_speed <- rbind.data.frame(lon_speed,c(ilimit,summary(rlmfit)$coefficients[2]*3600/180*pi)) # ra
    #print(c(rlmfit$fitted.values[1],last(rlmfit$fitted.values)))
    if (ilimit == 1) {
```

```
      #browser()
      plot(df$POSIX[idx],df$lon[idx],type="p",xlim=range(df$POSIX),ylim=range(df$lon,na.rm=T),xlab="Tim
      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.value
    }
    if (ilimit  > 1) {
      points(df$POSIX[idx],df$lon[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_right,

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

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

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

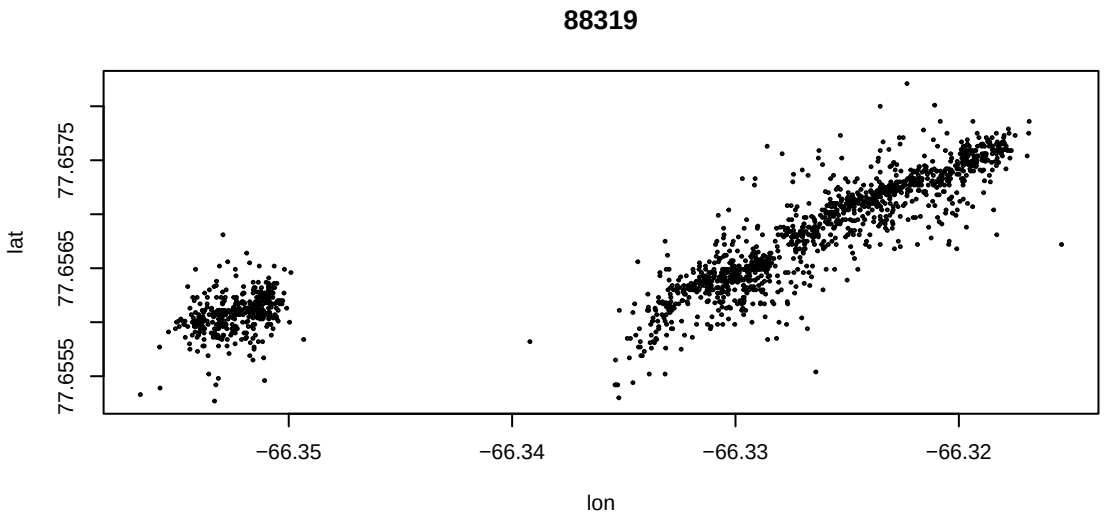
function to get interval and jump speeds

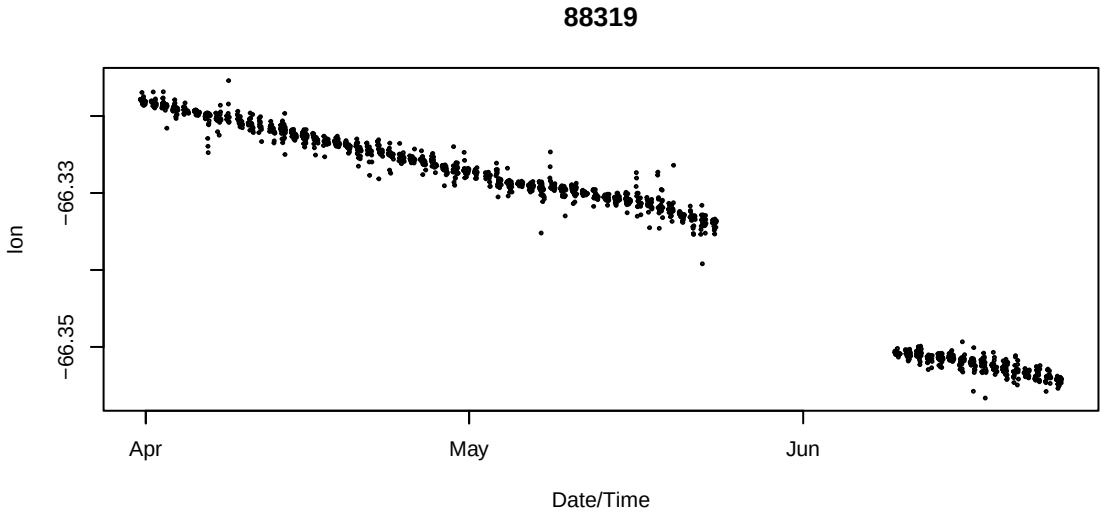
```
get_surge_speeds <- function(listerne,lon_in,lat_in)
{

  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))
}
```

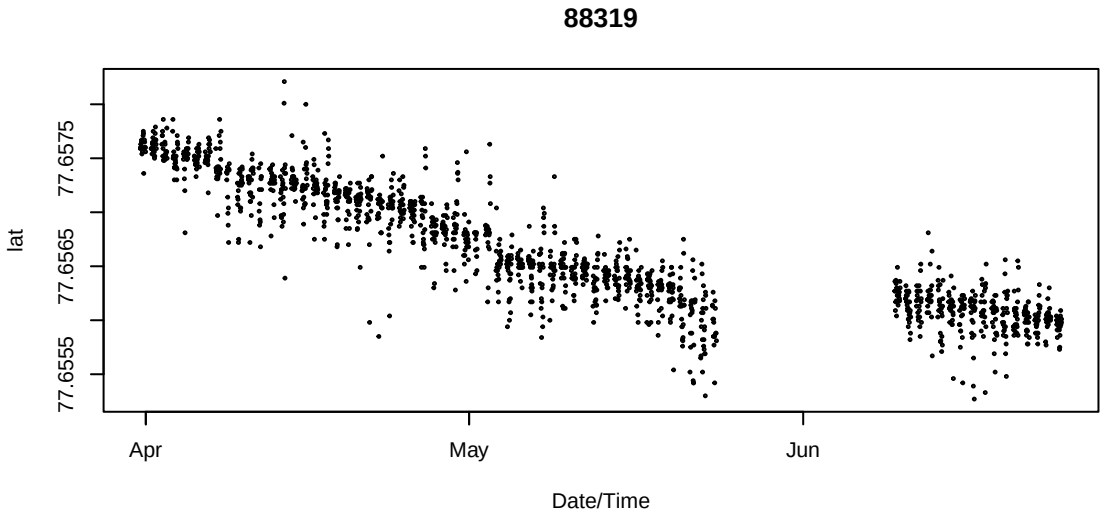
read and plot each file

```
# specify the important intervals:
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("2022-06-18 02:00:00",tz="UTC
)
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 (i_unitID in unitIDs)
#  for (i_unitID in c(88462))
{
  print("-----------------------------------------")
  print(paste(" Processing unitID ",i_unitID))
  mdx <- which(data$UnitID == i_unitID)
  df <- data[mdx,]
  plot_stuff(df,i_unitID,limitdates)

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

  segspeeds <- round(listerne$segspeed,2)
  print("Segment speeds in m/hr : ")
  print(segspeeds)
  # get surge speeds
  speeds <- get_surge_speeds(listerne,lon=median(df$lon,na.rm=T),lat=median(df$lat,na.rm=T))
  print(speeds)

}
```

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



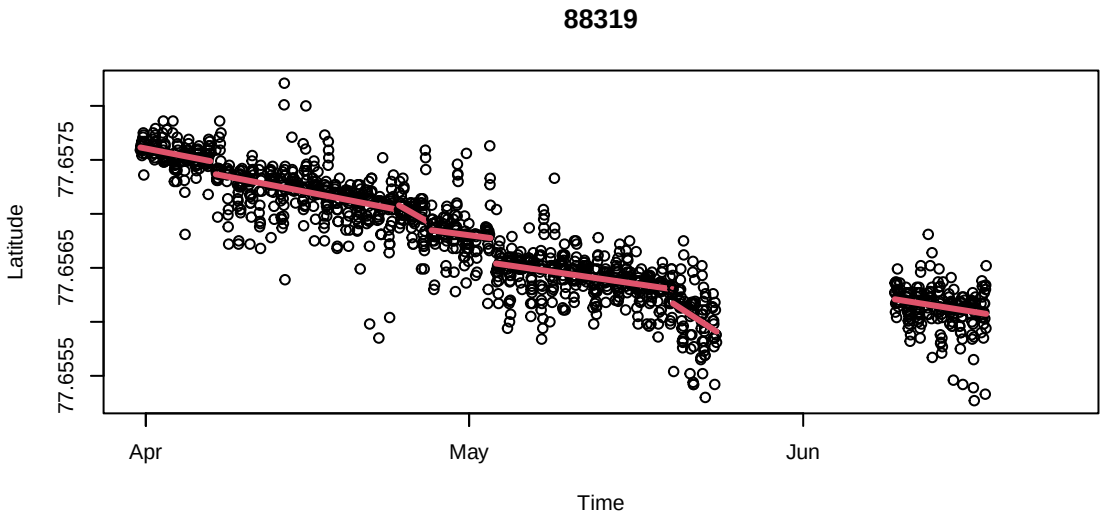


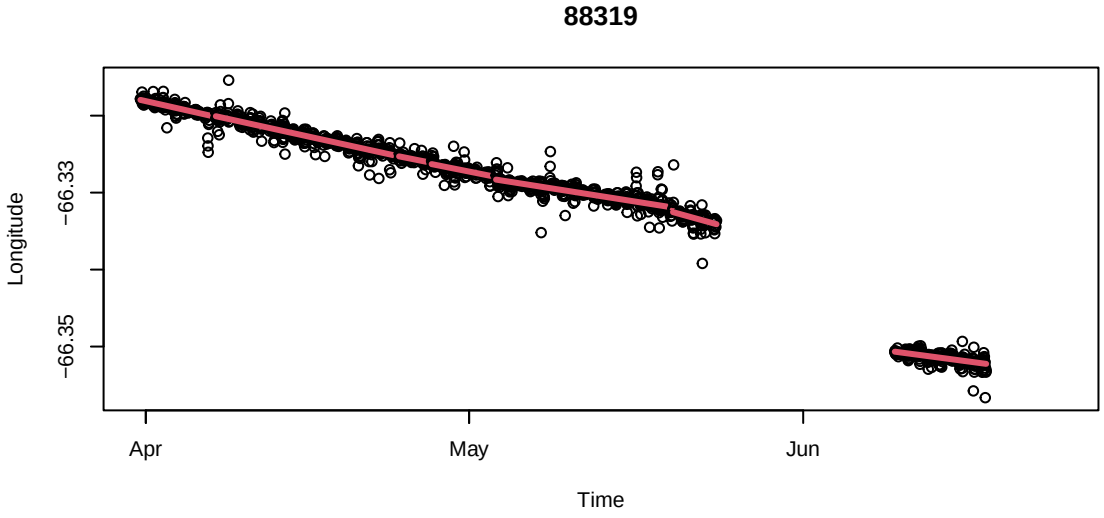



```
## [1] "Returning from plot_stuff"
```

```
## [1] "e"   "168"
```

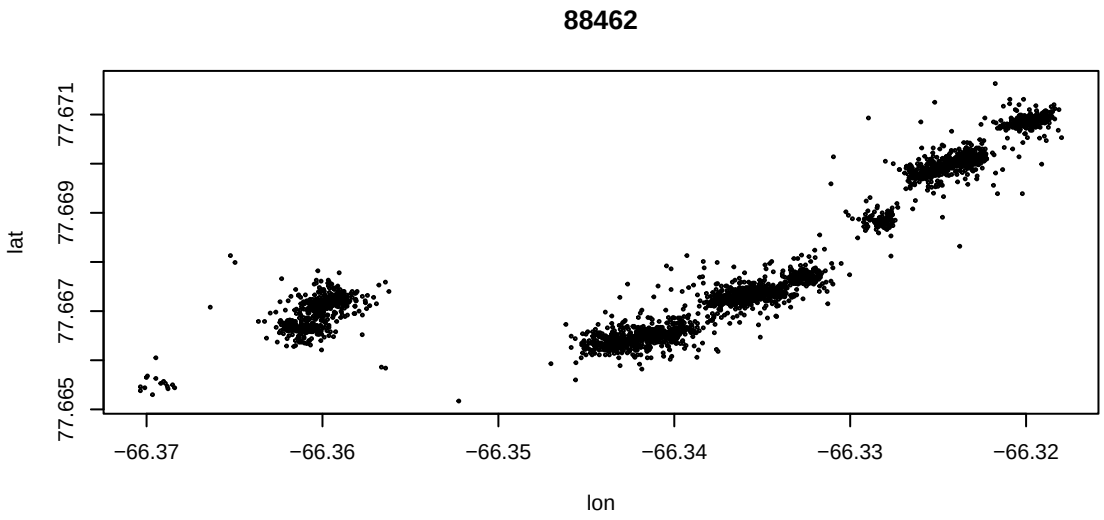
```
## [1] "e"   "385"
## [1] "e"  "73"
## [1] "e"   "131"
## [1] "e"   "404"
## [1] "e"  "92"
## [1] "e"   "219"
## [1] "f"   "168"
```

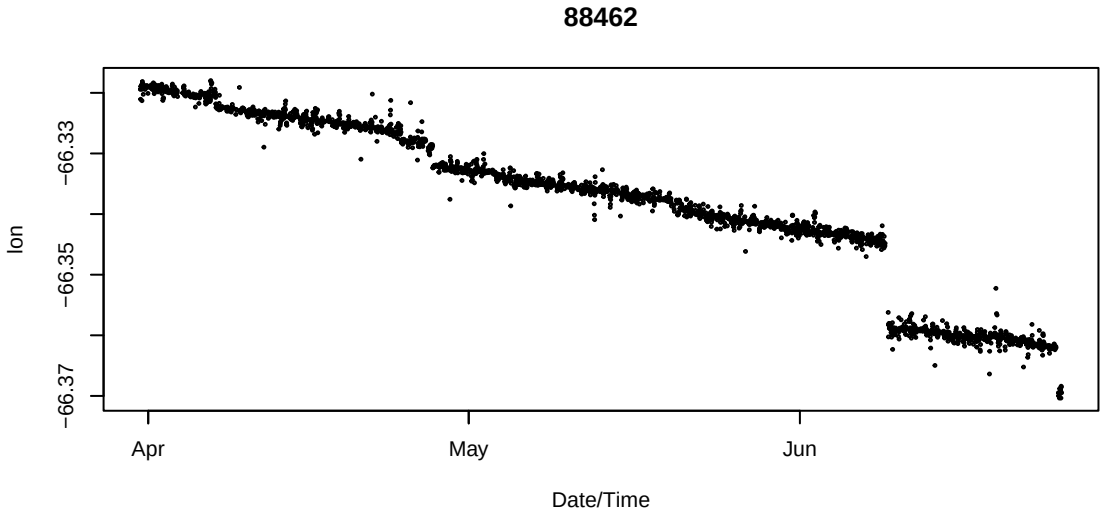


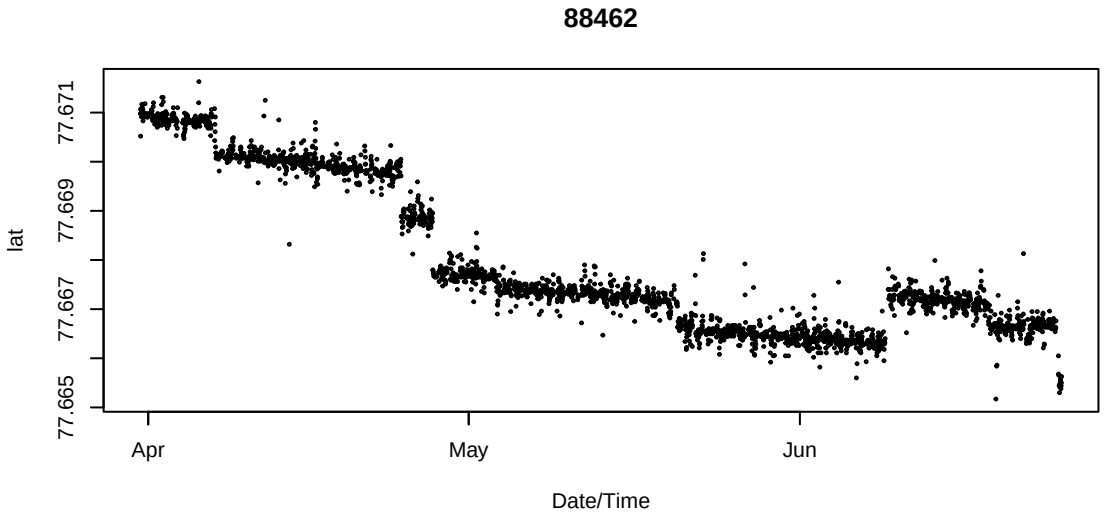



```
## [1] "f"   "385"
## [1] "f"  "73"
## [1] "f"   "131"
## [1] "f"   "404"
## [1] "f"  "92"
## [1] "f"   "219"
## [1] "Segment speeds in m/hr : "
## [1] 0.33 0.32 0.38 0.28 0.23 0.49 0.20
```

```
## $speed_jump
##   jump_number speed_metersph
## 1           1      181.96929
## 2           2      152.62560
## 3           3       68.86569
## 4           4      147.02680
## 5           5      154.55108
## 6           6      468.71376
##
## [1] "-----------------------------------------"
## [1] " Processing unitID  88462"
```



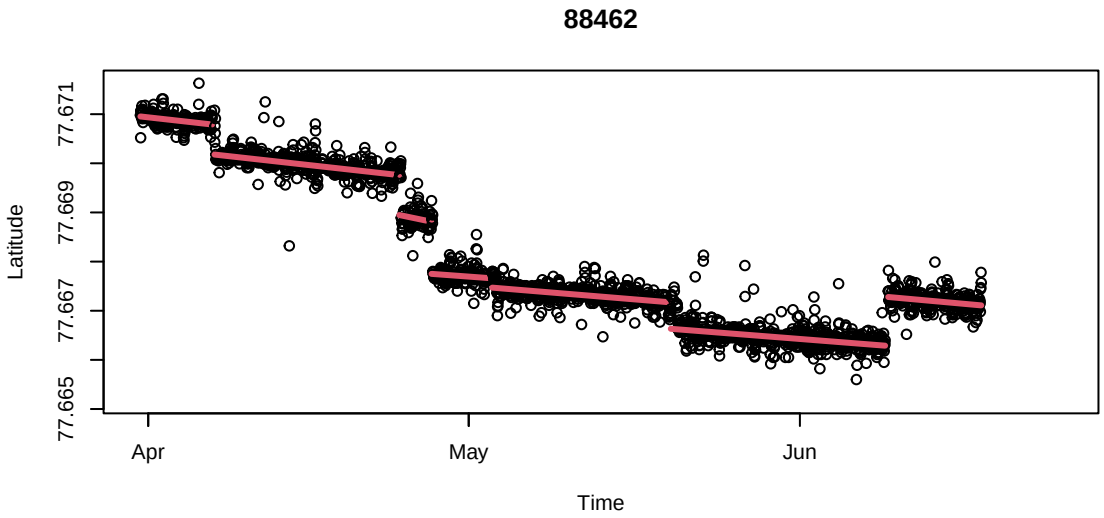


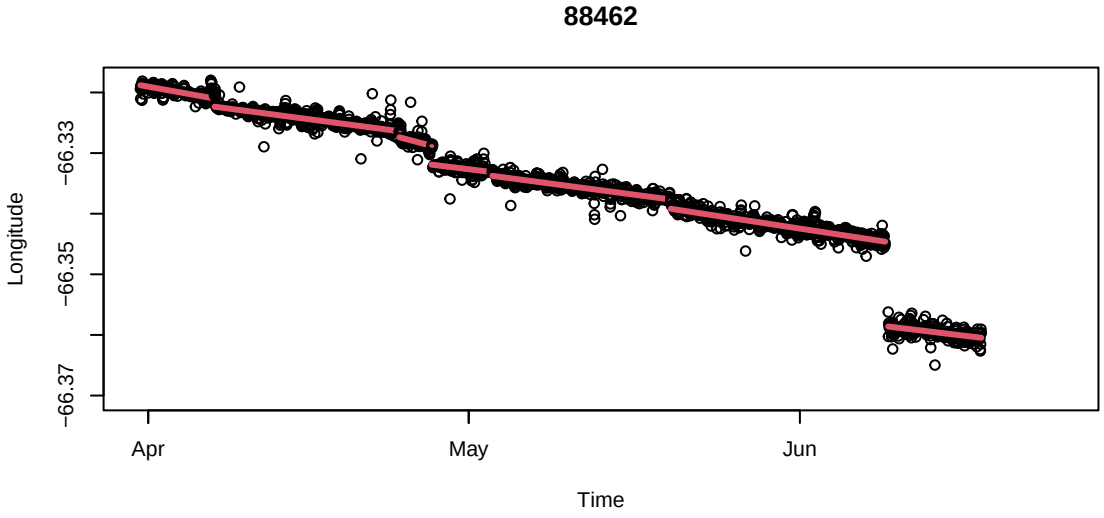



```
## [1] "Returning from plot_stuff"
```

```
## [1] "e"   "245"
```

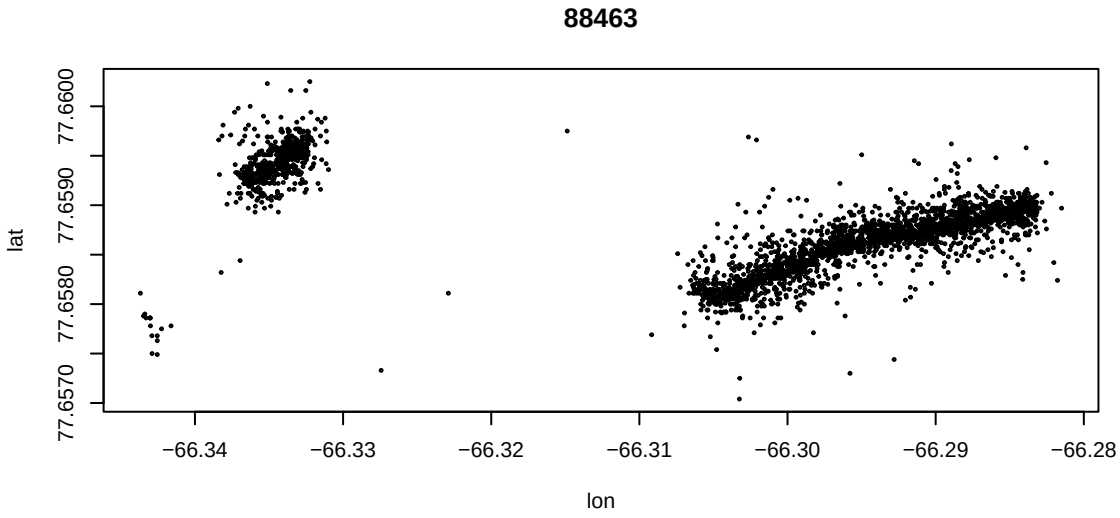
```
## [1] "e"   "609"
## [1] "e"   "115"
## [1] "e"   "213"
## [1] "e"   "615"
## [1] "e"   "744"
## [1] "e"   "338"
## [1] "f"   "245"
```

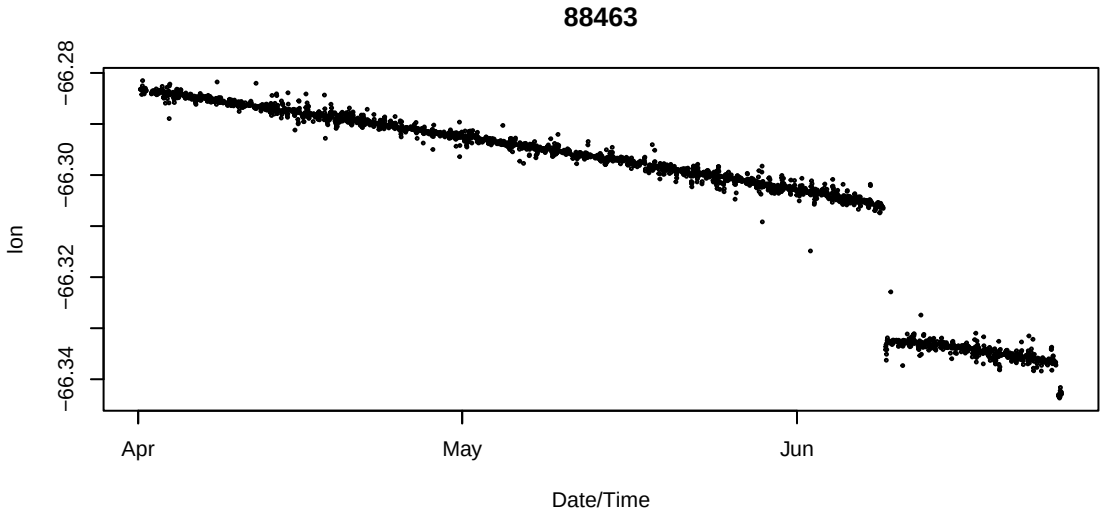


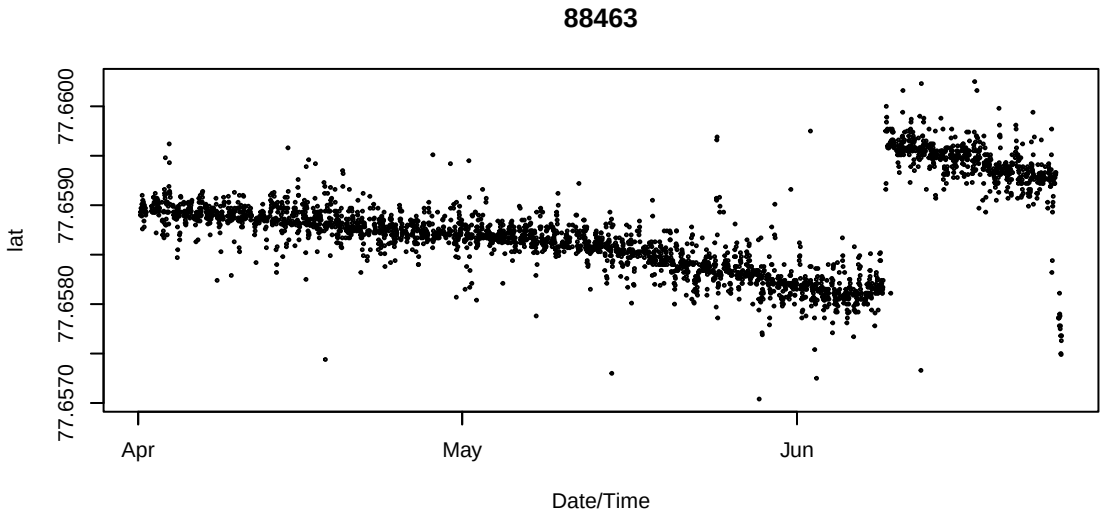



```
## [1] "f"   "609"
## [1] "f"   "115"
## [1] "f"   "213"
## [1] "f"   "615"
## [1] "f"   "744"
## [1] "f"   "338"
## [1] "Segment speeds in m/hr : "
## [1] 0.33 0.26 0.51 0.23 0.25 0.28 0.23
```

```
## $speed_jump
##   jump_number speed_metersph
## 1           1       223.5624
## 2           2       215.5637
## 3           3       196.8133
## 4           4       150.9626
## 5           5       289.2822
## 6           6       508.1187
##
## [1] "-----------------------------------------"
## [1] " Processing unitID  88463"
```



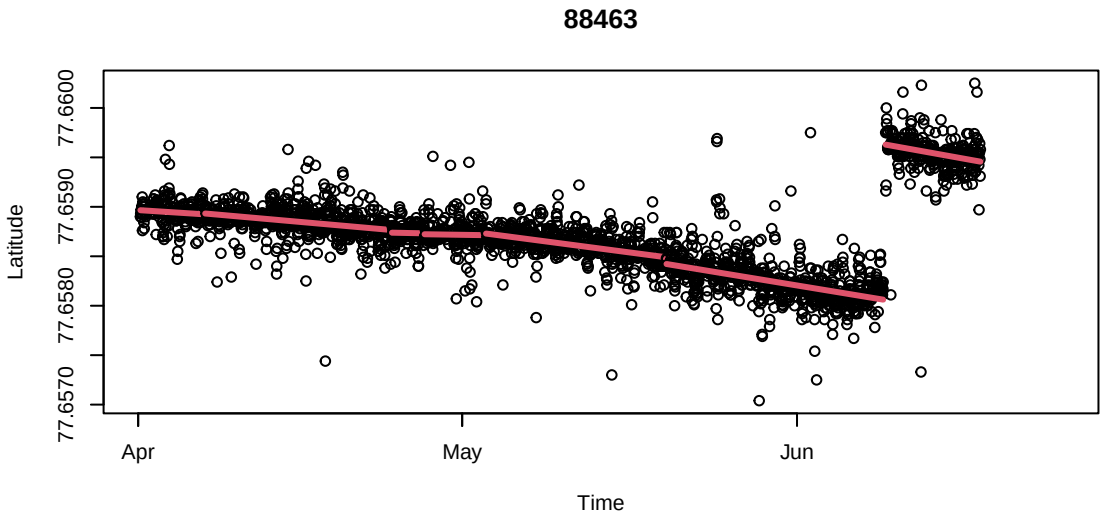


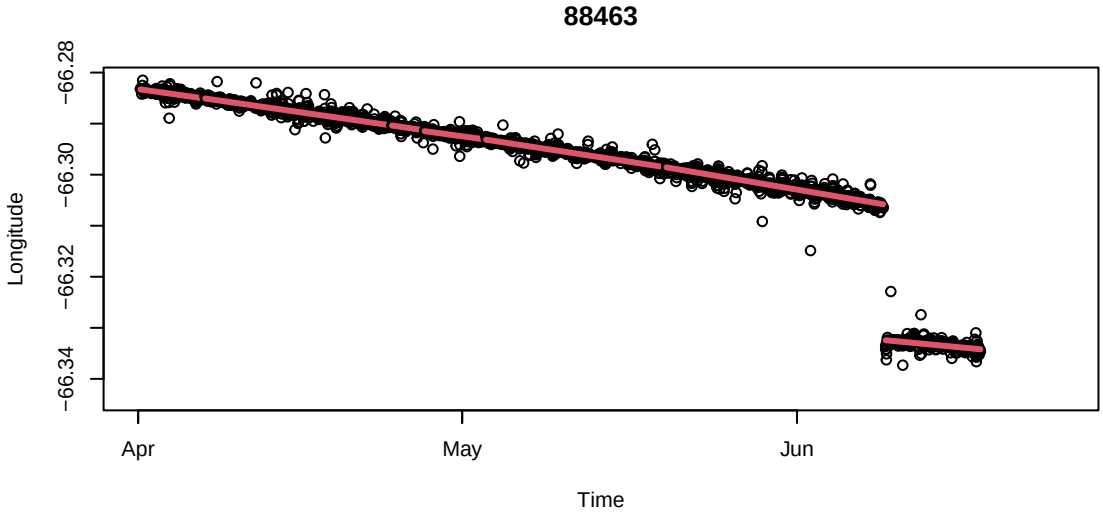



```
## [1] "Returning from plot_stuff"
```

```
## [1] "e"   "216"
```

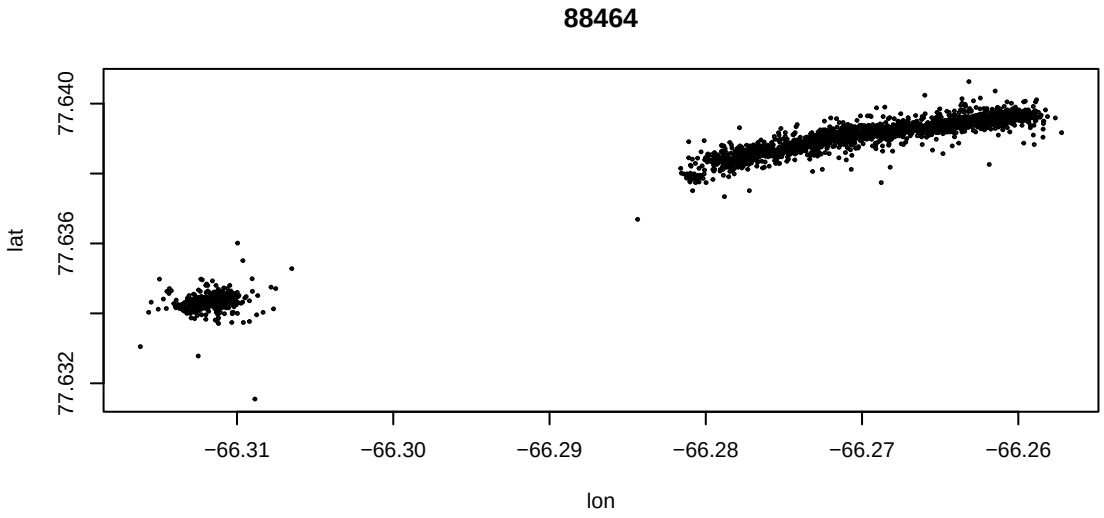
```
## [1] "e"   "633"
## [1] "e"   "103"
## [1] "e"   "201"
## [1] "e"   "615"
## [1] "e"   "764"
## [1] "e"   "341"
## [1] "f"   "216"
```

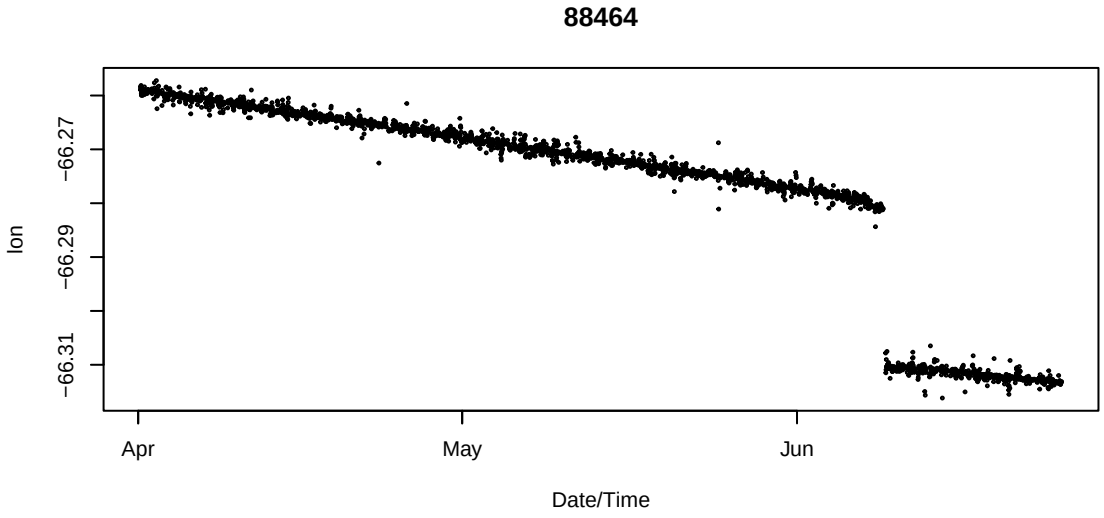


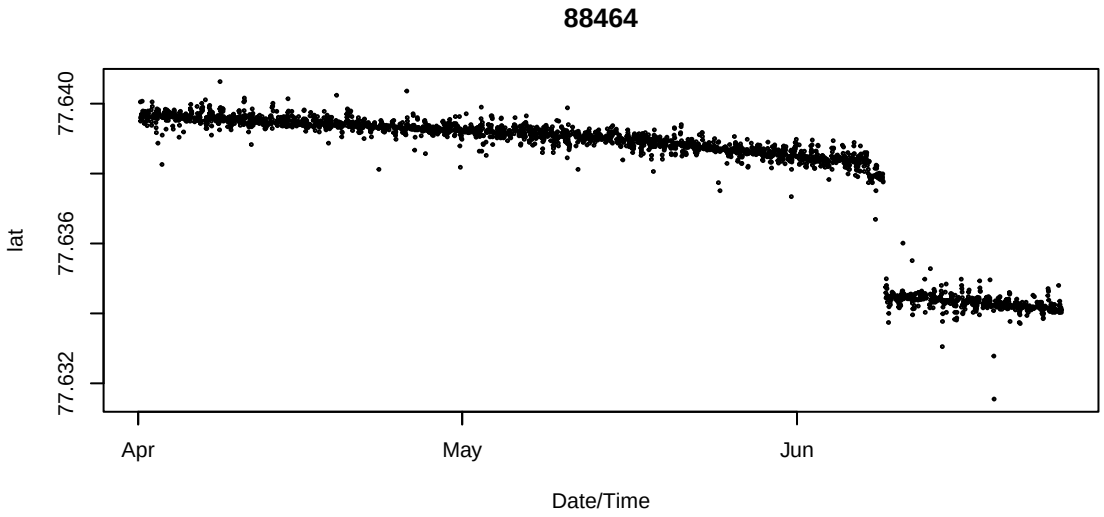



```
## [1] "f"   "633"
## [1] "f"   "103"
## [1] "f"   "201"
## [1] "f"   "615"
## [1] "f"   "764"
## [1] "f"   "341"
## [1] "Segment speeds in m/hr : "
## [1] 0.30 0.31 0.31 0.29 0.33 0.37 0.22
```

```
## $speed_jump
##   jump_number speed_metersph
## 1           1      169.33253
## 2           2      150.02906
## 3           3       63.91377
## 4           4      170.60416
## 5           5      309.75220
## 6           6      854.36439
##
## [1] "-----------------------------------------"
## [1] " Processing unitID  88464"
```



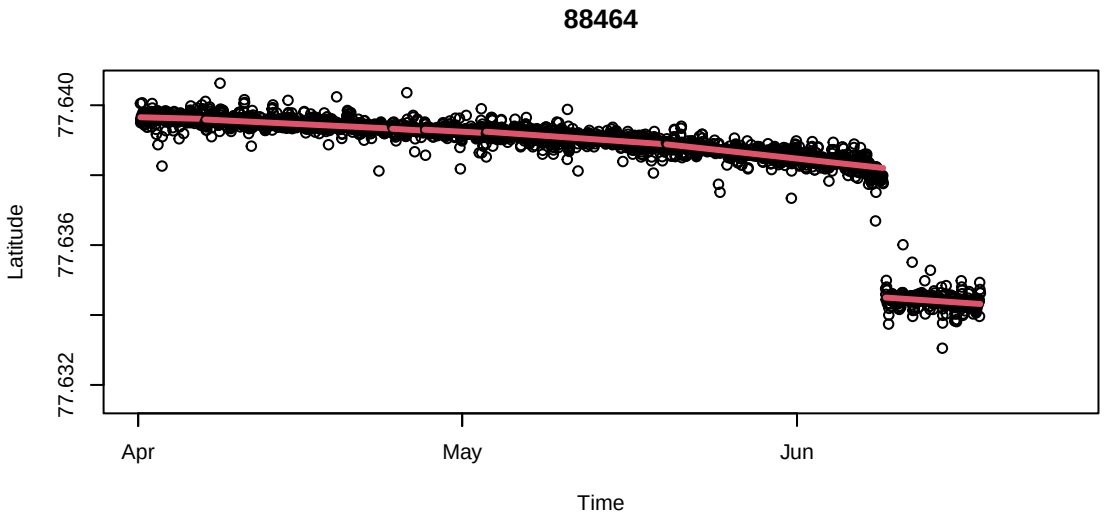


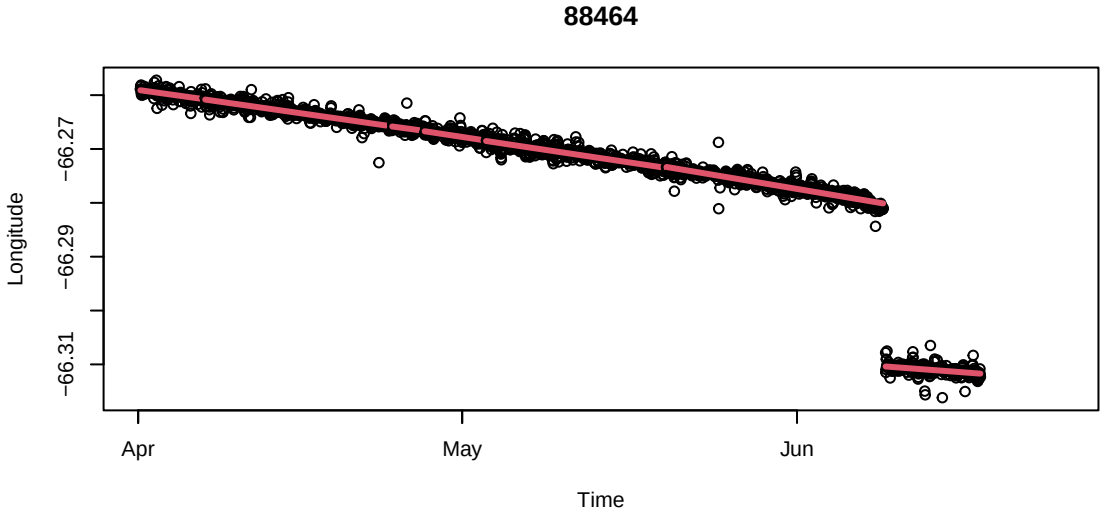



```
## [1] "Returning from plot_stuff"
```

```
## [1] "e"   "228"
```

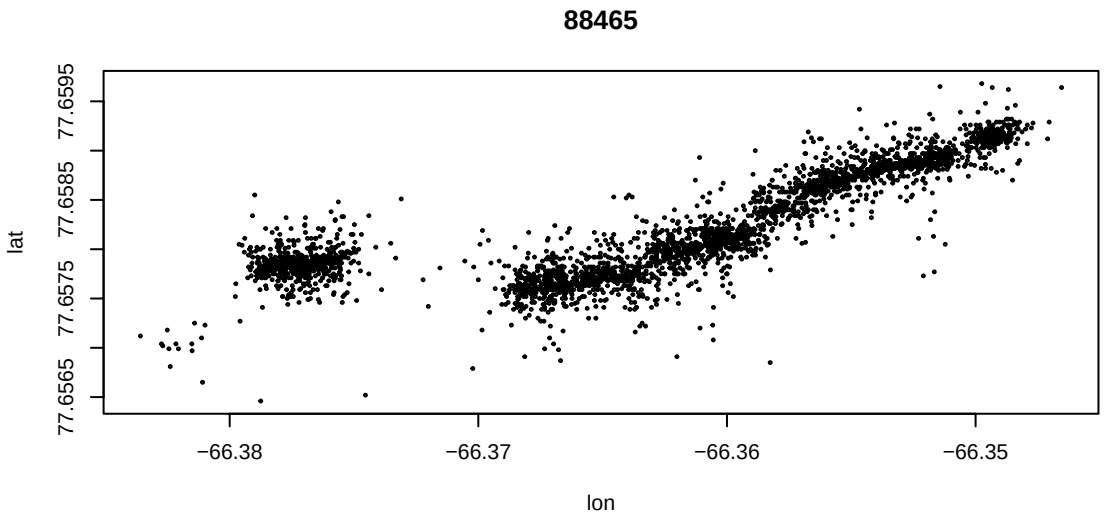
```
## [1] "e"   "631"
## [1] "e"   "114"
## [1] "e"   "213"
## [1] "e"   "615"
## [1] "e"   "746"
## [1] "e"   "338"
## [1] "f"   "228"
```

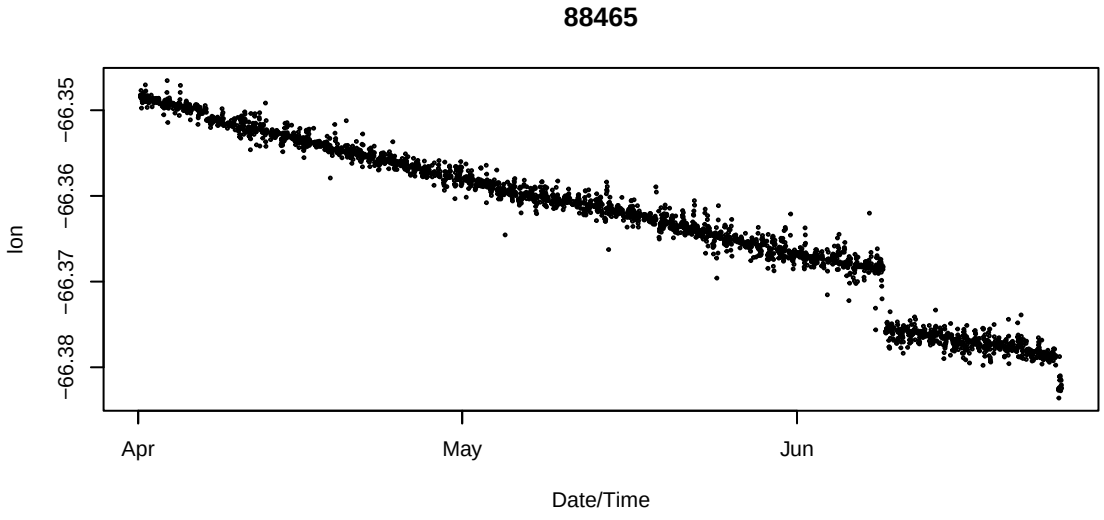


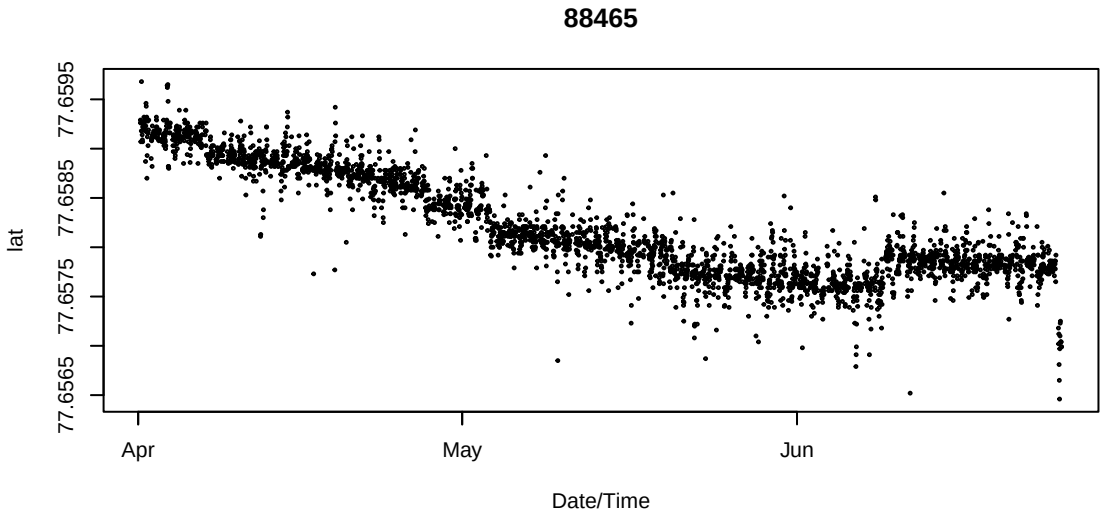



```
## [1] "f"   "631"
## [1] "f"   "114"
## [1] "f"   "213"
## [1] "f"   "615"
## [1] "f"   "746"
## [1] "f"   "338"
## [1] "Segment speeds in m/hr : "
## [1] 0.28 0.29 0.28 0.31 0.31 0.37 0.18
```

```
## $speed_jump
##   jump_number speed_metersph
## 1           1      161.74041
## 2           2      142.79304
## 3           3       62.00422
## 4           4      168.72309
## 5           5      299.02521
## 6           6     1045.06646
##
## [1] "-----------------------------------------"
## [1] " Processing unitID  88465"
```



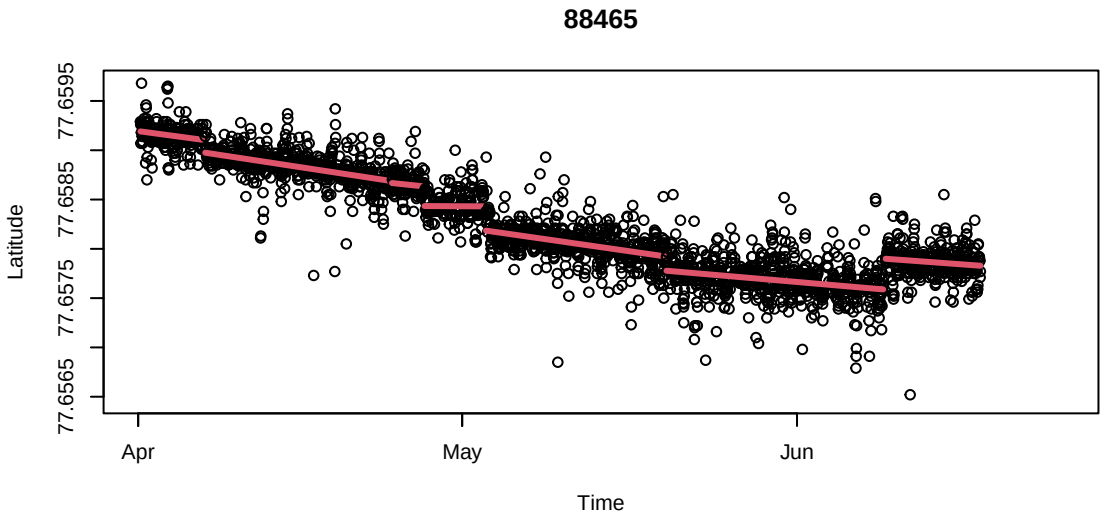


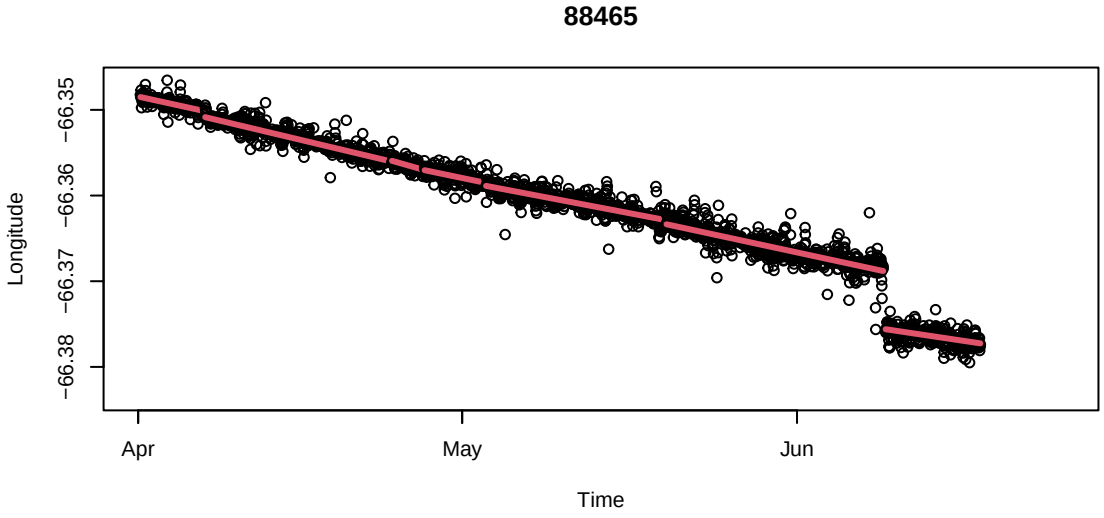



```
## [1] "Returning from plot_stuff"
```

```
## [1] "e"   "228"
```

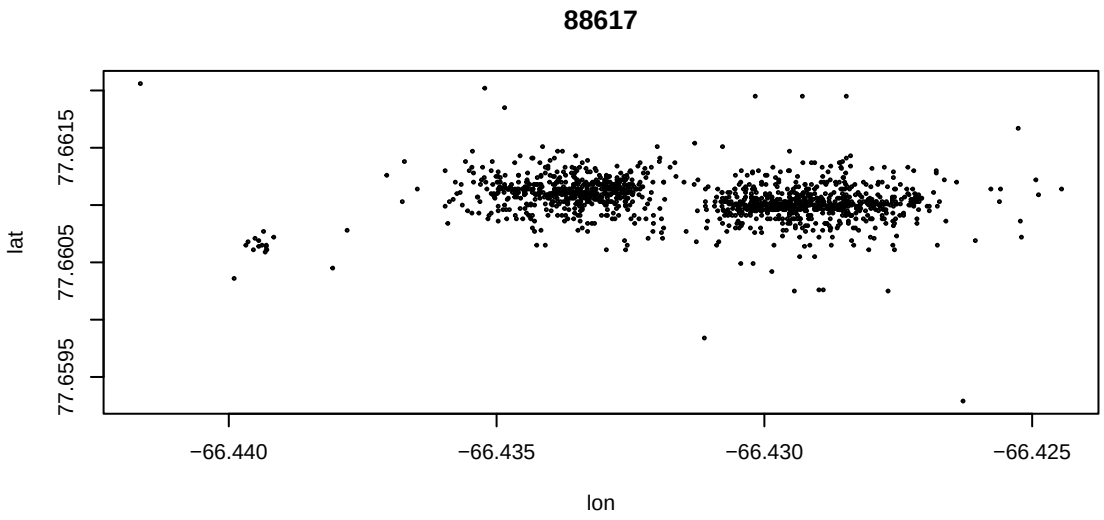
```
## [1] "e"   "654"
## [1] "e"   "115"
## [1] "e"   "195"
## [1] "e"   "612"
## [1] "e"   "749"
## [1] "e"   "344"
## [1] "f"   "228"
```

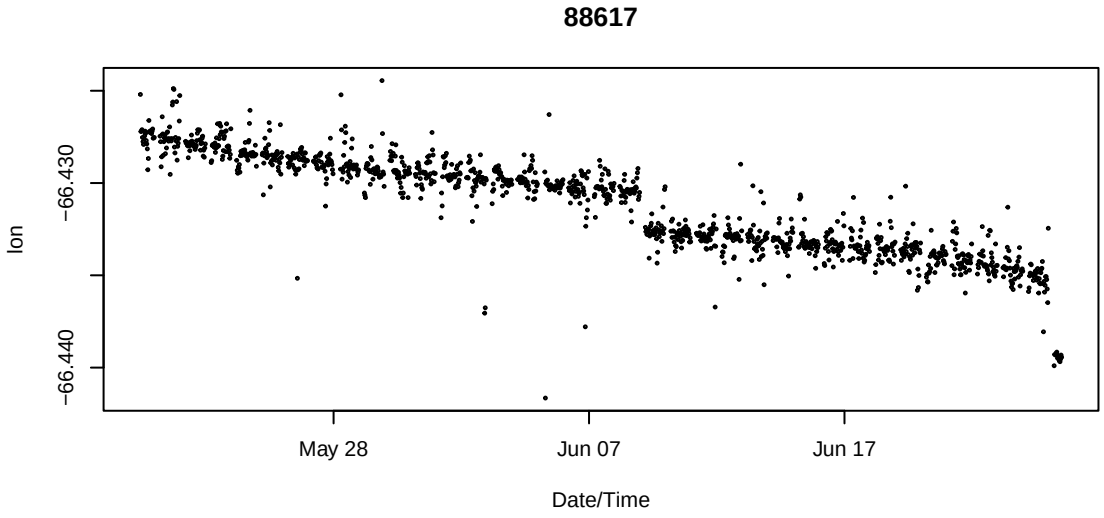


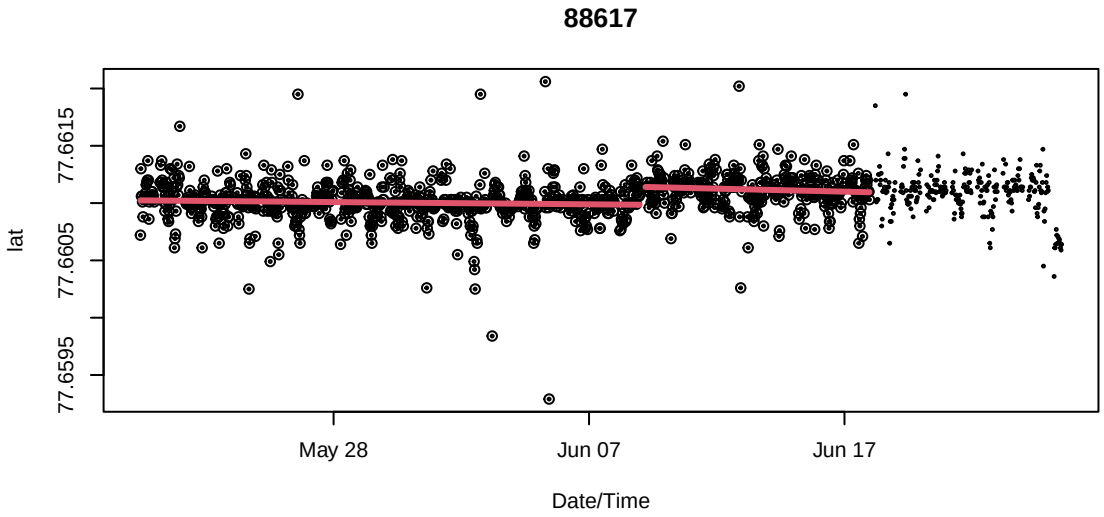



```
## [1] "f"   "654"
## [1] "f"   "115"
## [1] "f"   "195"
## [1] "f"   "612"
## [1] "f"   "749"
## [1] "f"   "344"
## [1] "Segment speeds in m/hr : "
## [1] 0.28 0.31 0.36 0.26 0.25 0.27 0.19
```

```
## $speed_jump
##   jump_number speed_metersph
## 1           1      187.53361
## 2           2      151.52729
## 3           3       65.46887
## 4           4      149.96311
## 5           5      245.30925
## 6           6      330.40425
##
## [1] "----------------------------------------"
## [1] " Processing unitID  88617"
```








```
## [1] "Returning from plot_stuff"
```

```
## [1] "e"   "743"
## [1] "e"   "344"
## [1] "f"   "743"
## [1] "f"   "344"
## [1] "Segment speeds in m/hr : "
## [1] 0.15 0.11
## $speed_jump
##   jump_number speed_metersph
## 1           1       139.5514
```

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

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