E3Task

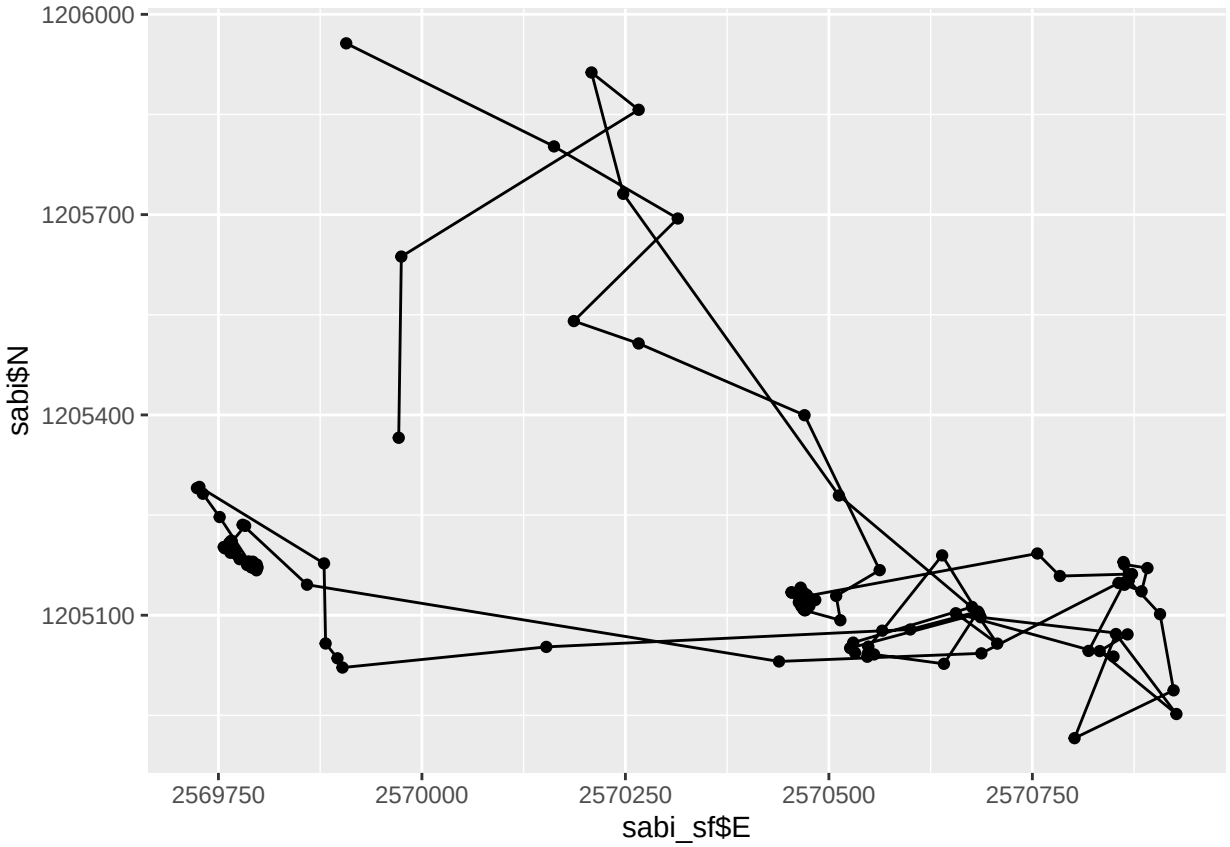
Author: Yelu He

2023-05-10

Library preparation

Task 0: Import data and preparation

```
## Load the data
wildschwein <- read_delim("wildschwein_BE_2056.csv", ",")

## select data of sabi in a specific time range
sabi <- wildschwein %>%
    filter(TierName == "Sabi", DatetimeUTC >= "2015-07-01", DatetimeUTC < "2015-07-03")

## View the map of sabi
sabi_sf <- st_as_sf(sabi, coords = c("E", "N"), crs = 2056, remove = FALSE)
ggplot(sabi_sf, aes(x = sabi_sf$E,
                    y = sabi$N)) +
  geom_point() +
  geom_path()
```

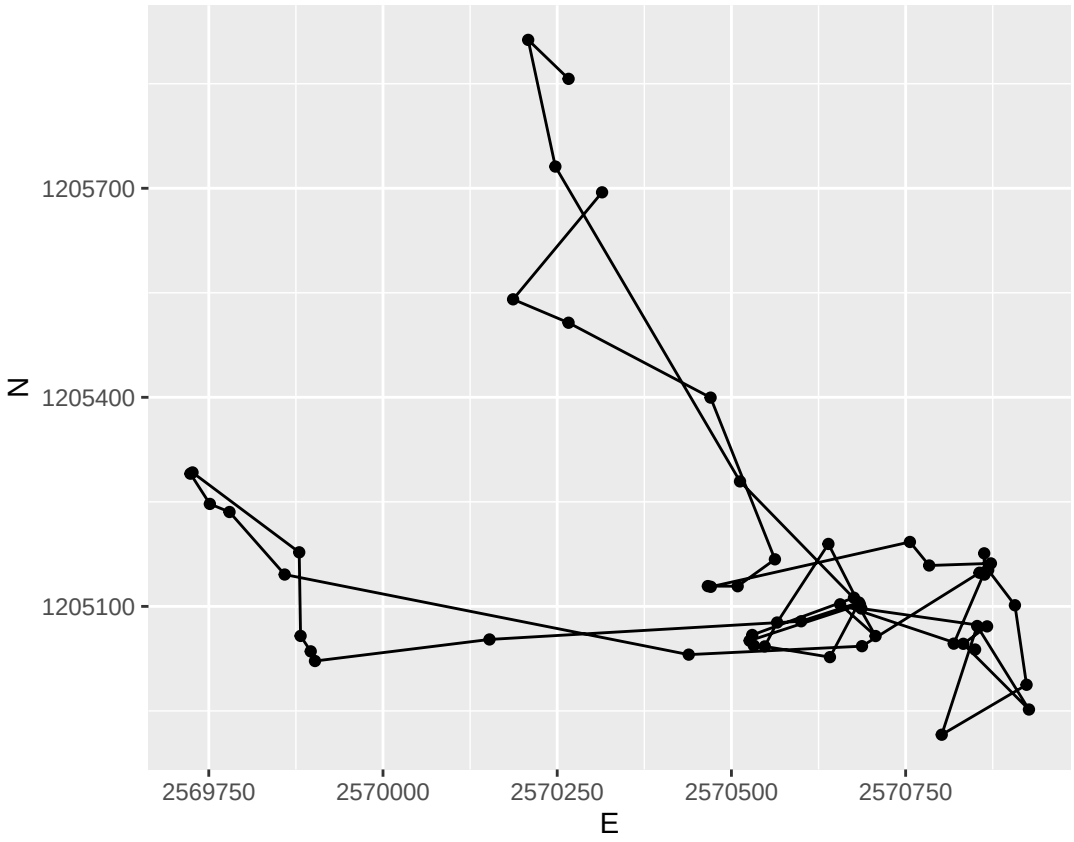



```
## Specify a temporal window
## Measure distance from every point to every other point within the temporal window
sabi <- sabi %>%
  mutate(
        nMinus2 = sqrt((lag(E, 2) - E)^2 + (lag(N, 2) - N)^2), # distance to pos -30 minutes
        nMinus1 = sqrt((lag(E, 1) - E)^2 + (lag(N, 1) - N)^2), # distance to pos -15 minutes
        nPlus1  = sqrt((E - lead(E, 1))^2 + (N - lead(N, 1))^2), # distance to pos +15 mintues
        nPlus2  = sqrt((E - lead(E, 2))^2 + (N - lead(N, 2))^2) # distance to pos +30 minutes
    )
sabi <- sabi |>
    rowwise() |>
    mutate(
        stepMean = mean(c(nMinus2, nMinus1, nPlus1, nPlus2))
    ) |>
    ungroup()
sabi
```

```
## # A tibble: 192 x 11
##    TierID TierName CollarID DatetimeUTC                E       N nMinus2 nMinus1
##    <chr>  <chr>       <dbl> <dttm>                 <dbl>   <dbl>   <dbl>   <dbl>
##  1 002A   Sabi        12275 2015-06-30 22:00:13 2569972.  1.21e6   NA      NA
##  2 002A   Sabi        12275 2015-06-30 22:16:06 2569975.  1.21e6   NA     271.
##  3 002A   Sabi        12275 2015-06-30 22:30:19 2570266.  1.21e6  573.    365.
##  4 002A   Sabi        12275 2015-06-30 22:45:13 2570208.  1.21e6  361.     80.5
##  5 002A   Sabi        12275 2015-06-30 23:00:10 2570247.  1.21e6  127.    186.
##  6 002A   Sabi        12275 2015-06-30 23:15:17 2570512.  1.21e6  703.    524.
```

```
##  7 002A   Sabi        12275 2015-06-30 23:30:38 2570684.  1.21e6  766.    247.
##  8 002A   Sabi        12275 2015-06-30 23:45:16 2570526.  1.21e6  229.    167.
##  9 002A   Sabi        12275 2015-07-01 00:00:10 2570532.  1.21e6  163.      9.33
## 10 002A   Sabi        12275 2015-07-01 00:15:14 2570530.  1.21e6    8.98   15.4
## # i 182 more rows
## # i 3 more variables: nPlus1 <dbl>, nPlus2 <dbl>, stepMean <dbl>
```

```
## Remove static points
sabi <- sabi |>
    ungroup() |>
    mutate(static = stepMean < mean(stepMean, na.rm = TRUE))

sabi_filter <- sabi |>
    filter(!static)

sabi_filter |>
    ggplot(aes(E, N)) +
    geom_path() +
    geom_point() +
    coord_fixed() +
    theme(legend.position = "bottom")
```




```
# Preparation
## Load the data
move0 <- read_delim("movementdata/yelu_dataset_01.csv", ";")
```

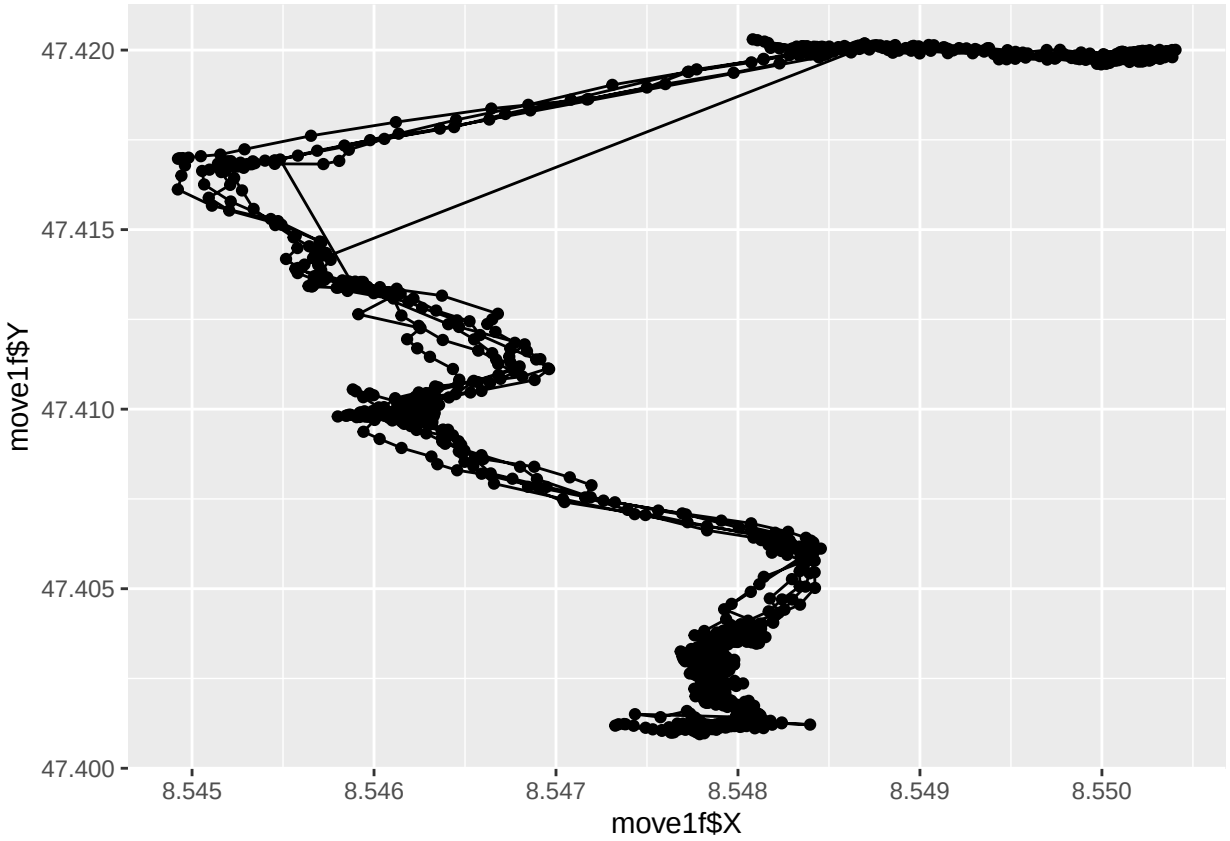
```r
## Use right coordinate system and preserve original E/N columns
move0 <- st_as_sf(move0, coords = c("Longitude", "Latitude"), crs = 2056, remove = FALSE)
## Select needed columns
move1 <- select(move0, Date, Time, Latitude, Longitude, geometry)
head(move1)
```

```
## Simple feature collection with 6 features and 4 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: 8.678872 ymin: 47.21852 xmax: 8.679217 ymax: 47.2186
## Projected CRS: CH1903+ / LV95
## # A tibble: 6 x 5
##   Date       Time     Latitude Longitude            geometry
##   <chr>      <time>      <dbl>     <dbl>         <POINT [m]>
## 1 21.04.2023 14:13:35     47.2      8.68 (8.679217 47.21852)
## 2 21.04.2023 14:13:40     47.2      8.68 (8.679217 47.21852)
## 3 21.04.2023 14:13:45     47.2      8.68 (8.679094 47.21855)
## 4 21.04.2023 14:13:50     47.2      8.68 (8.679023 47.21856)
## 5 21.04.2023 14:13:55     47.2      8.68 (8.678975 47.21858)
## 6 21.04.2023 14:14:00     47.2      8.68  (8.678872 47.2186)
```

```r
move1_coordinates <- st_coordinates(move1)
move1 <- cbind(move1, move1_coordinates)
move1f <- move1[move1$Date == '25.04.2023',]
```

Task 1: Segmentation

```r
## View the map of records
ggplot(move1f,
       aes(x = move1f$X,
           y = move1f$Y)) +
  geom_point() +
  geom_path()
```

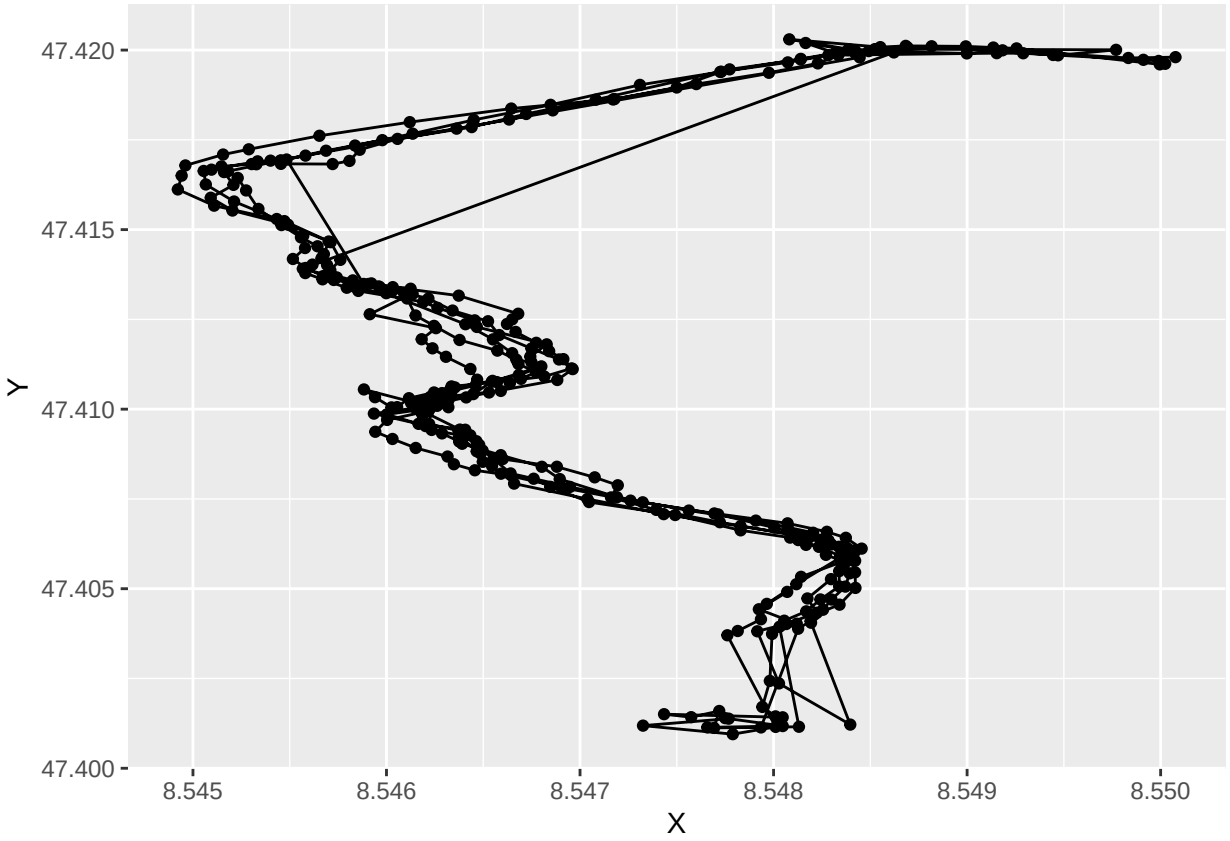



```
## Specify a temporal window
## Measure distance from every point to every other point within the temporal window
move1f <- move1f %>%
  mutate(
        nMinus2 = sqrt((lag(X, 2) - X)^2 + (lag(Y, 2) - Y)^2), # distance to pos -30 minutes
        nMinus1 = sqrt((lag(X, 1) - X)^2 + (lag(Y, 1) - Y)^2), # distance to pos -15 minutes
        nPlus1  = sqrt((X - lead(X, 1))^2 + (Y - lead(Y, 1))^2), # distance to pos +15 mintues
        nPlus2  = sqrt((X - lead(X, 2))^2 + (Y - lead(Y, 2))^2) # distance to pos +30 minutes
    )


move1f <- move1f %>%
    rowwise() %>%
    mutate(
        stepMean = mean(c(nMinus2, nMinus1, nPlus1, nPlus2))
    ) %>%
    ungroup()
move1f
```

```
## Simple feature collection with 1399 features and 11 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: 8.544922 ymin: 47.40095 xmax: 8.550404 ymax: 47.4203
## Projected CRS: CH1903+ / LV95
## # A tibble: 1,399 x 12
##    Date       Time     Latitude Longitude     X     Y            geometry
```

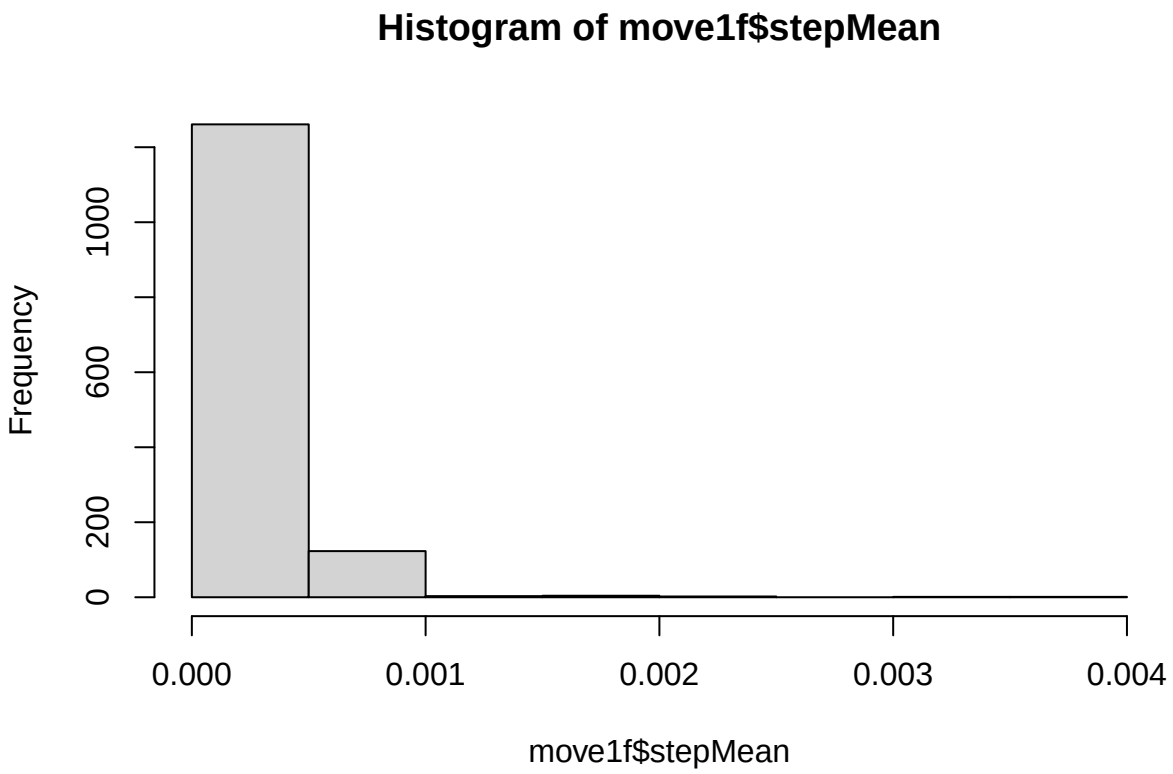
```
##    <chr>      <time>      <dbl>     <dbl> <dbl> <dbl>         <POINT [m]>
##  1 25.04.2023 13:03:50     47.4      8.55  8.55  47.4 (8.549048 47.42005)
##  2 25.04.2023 13:03:55     47.4      8.55  8.55  47.4 (8.548922 47.42006)
##  3 25.04.2023 13:04:00     47.4      8.55  8.55  47.4 (8.548797 47.42006)
##  4 25.04.2023 13:04:05     47.4      8.55  8.55  47.4 (8.548722 47.42002)
##  5 25.04.2023 13:04:10     47.4      8.55  8.55  47.4 (8.548958 47.42006)
##  6 25.04.2023 13:04:15     47.4      8.55  8.55  47.4 (8.548877 47.42005)
##  7 25.04.2023 13:04:20     47.4      8.55  8.55  47.4 (8.548759 47.42014)
##  8 25.04.2023 13:04:25     47.4      8.55  8.55  47.4 (8.548685 47.42015)
##  9 25.04.2023 13:04:30     47.4      8.55  8.55  47.4  (8.54866 47.42012)
## 10 25.04.2023 13:04:35     47.4      8.55  8.55  47.4 (8.548633 47.42007)
## # i 1,389 more rows
## # i 5 more variables: nMinus2 <dbl>, nMinus1 <dbl>, nPlus1 <dbl>, nPlus2 <dbl>,
## #   stepMean <dbl>
```

```r
## Remove static points
move1f <- move1f |>
    ungroup() |>
    mutate(static = stepMean < mean(stepMean, na.rm = TRUE))

move1f_filter <- move1f %>%
    filter(!static)

ggplot(data = move1f_filter, aes(X, Y)) +
    geom_path() +
    geom_point() +
    # coord_fixed() +
    theme(legend.position = "bottom")
```

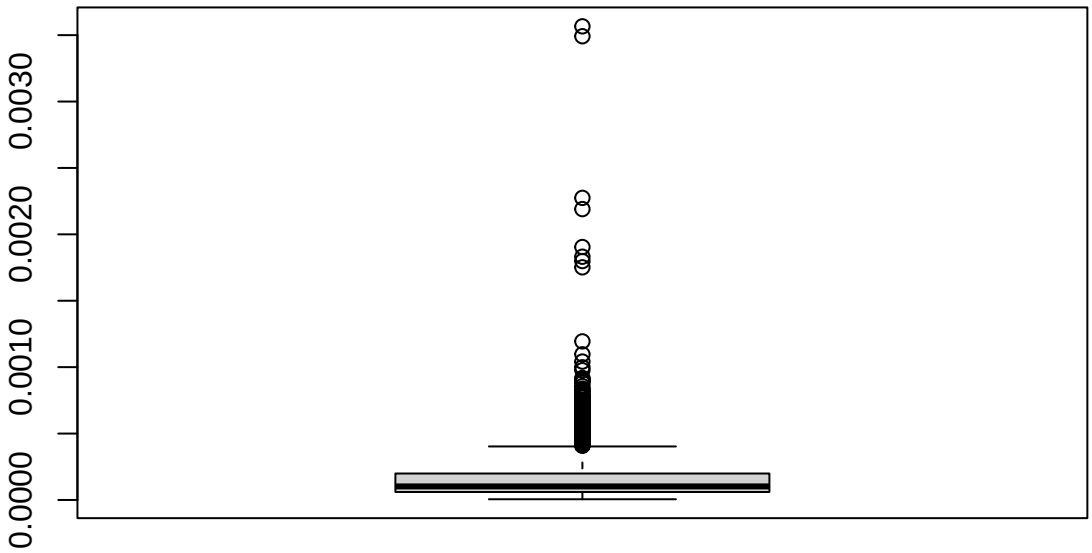


Task 2: Specify and apply threshold d

```
## Explore the value of stepMean
hist(move1f$stepMean)
```




```
boxplot(move1f$stepMean)
```



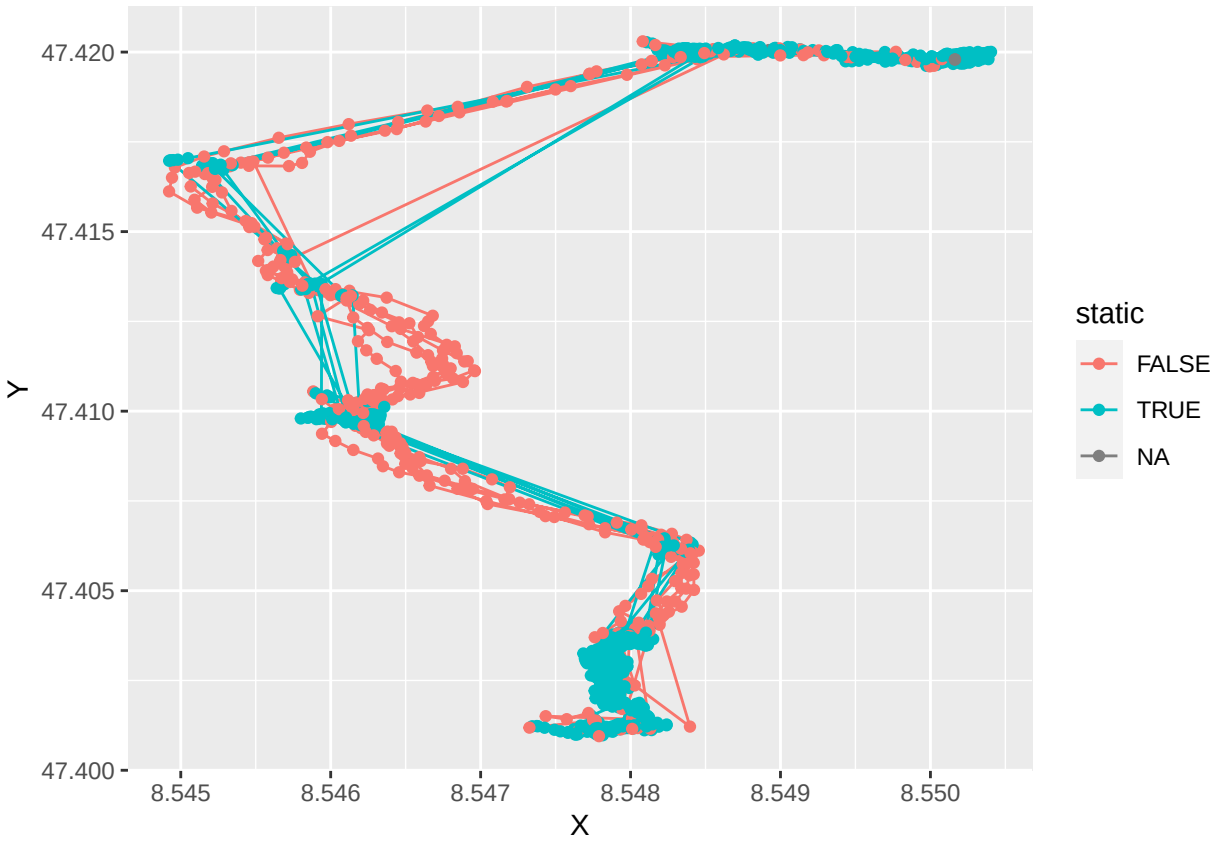

```
summary(move1f$stepMean)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.     NA's
## 0.000005 0.000060 0.000102 0.000189 0.000199 0.003566        4
```

```
## Use the mean of stepMean values as threshold
d <- mean(move1f$stepMean)
## Set to static column
# move1f <- move1f |>
#     ungroup() |>
#     mutate(static = stepMean < mean(stepMean,
#           na.rm = TRUE))
```

Task 3: Visualize segmented trajectories

```
ggplot(data = move1f, aes(X, Y, color = static)) +
    geom_path() +
    geom_point() +
    # coord_equal() +
    theme(legend.position = "right")
```



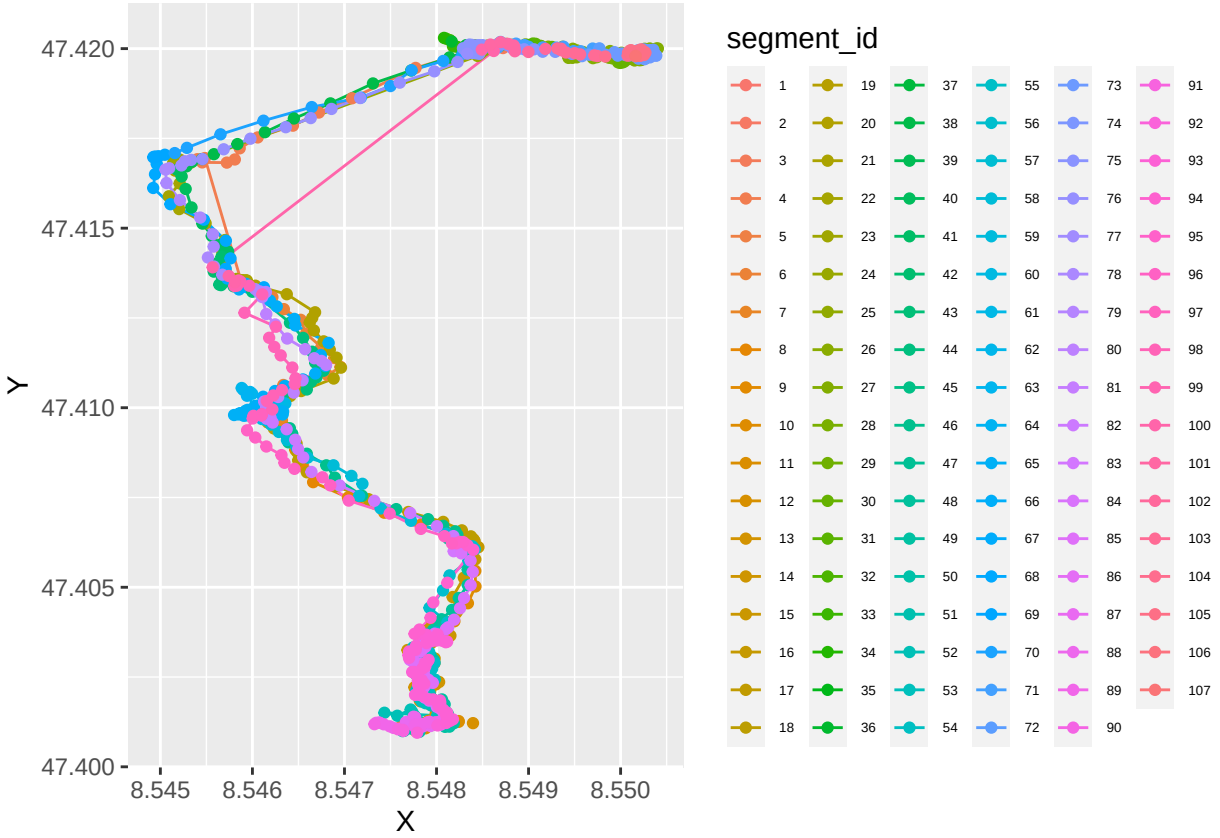
Task 4: Segment-based analysis

```
## Assign segment ID for each segment
rle_id <- function(vec) {
    x <- rle(vec)$lengths
    as.factor(rep(seq_along(x), times = x))
}
## Name it _f1 as _filter1, different from _filter
move1f_f1 <- move1f |>
    mutate(segment_id = rle_id(static))
head(move1f_f1)
```

```
## Simple feature collection with 6 features and 13 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: 8.548722 ymin: 47.42002 xmax: 8.549048 ymax: 47.42006
## Projected CRS: CH1903+ / LV95
## # A tibble: 6 x 14
##   Date       Time     Latitude Longitude     X     Y            geometry
##   <chr>      <time>      <dbl>     <dbl> <dbl> <dbl>         <POINT [m]>
## 1 25.04.2023 13:03:50     47.4      8.55  8.55  47.4 (8.549048 47.42005)
## 2 25.04.2023 13:03:55     47.4      8.55  8.55  47.4 (8.548922 47.42006)
## 3 25.04.2023 13:04:00     47.4      8.55  8.55  47.4 (8.548797 47.42006)
## 4 25.04.2023 13:04:05     47.4      8.55  8.55  47.4 (8.548722 47.42002)
```

```
## 5 25.04.2023 13:04:10     47.4      8.55  8.55  47.4 (8.548958 47.42006)
## 6 25.04.2023 13:04:15     47.4      8.55  8.55  47.4 (8.548877 47.42005)
## # i 7 more variables: nMinus2 <dbl>, nMinus1 <dbl>, nPlus1 <dbl>, nPlus2 <dbl>,
## #   stepMean <dbl>, static <lgl>, segment_id <fct>
```

```
## Plot the segments based on segment ID
ggplot(data = move1f_f1, aes(X, Y, color = segment_id)) +
    geom_path() +
    geom_point() +
    # coord_equal() +
    theme(legend.position = "right", legend.key.size = unit(0.5,'cm'), legend.text = element_text(size =
```



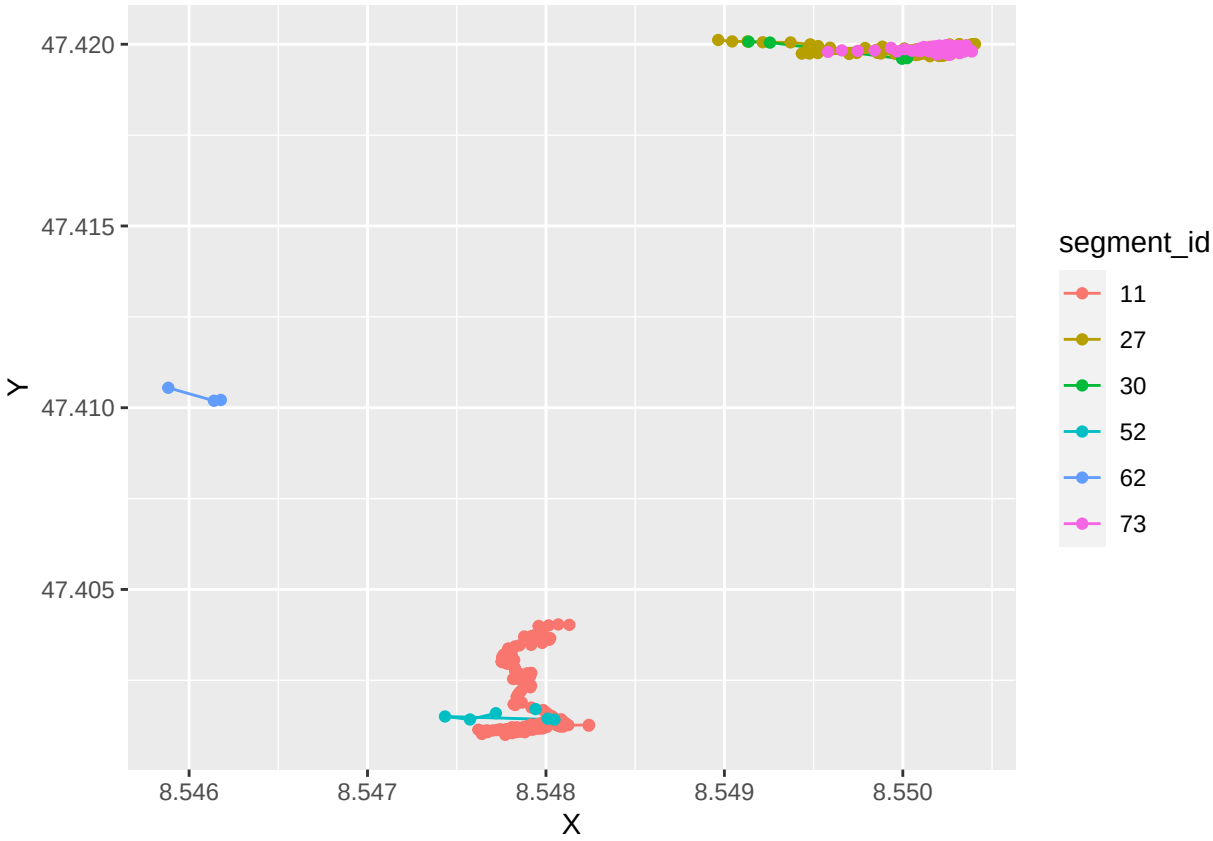

```
## Calculate the duration time for each segment
move1f_f1 <- move1f_f1 %>% group_by(segment_id) %>% mutate(duration = max(Time) - min(Time))
## Select the segments which has a duration more than five minutes
move1f_f1_5min <- move1f_f1[move1f_f1$duration >= 300,]
unique(move1f_f1_5min$segment_id)
```

```
## [1] 11 27 30 52 62 73
## 107 Levels: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 ... 107
```

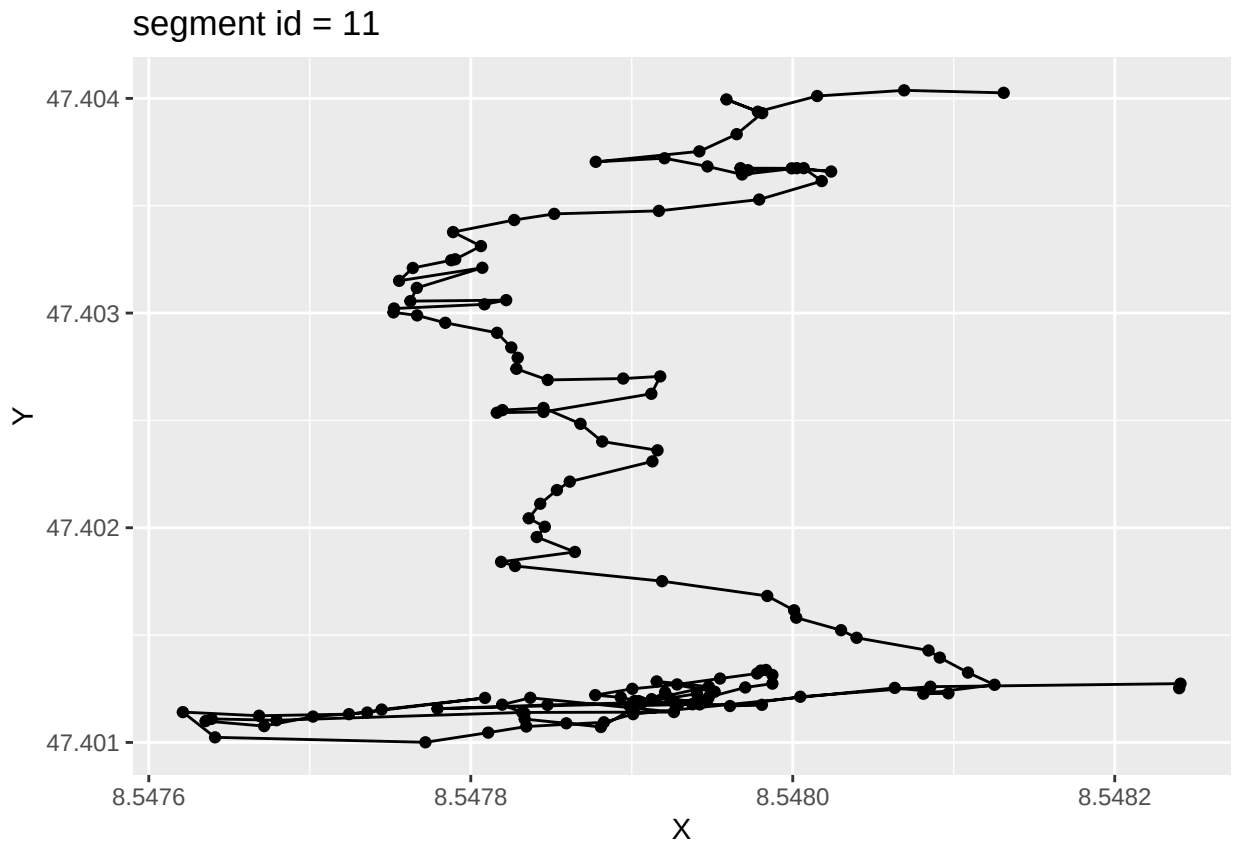
```
unique(move1f_f1_5min$duration)
```

```
## [1] 1130 3625 7425 1115  575 1500
```

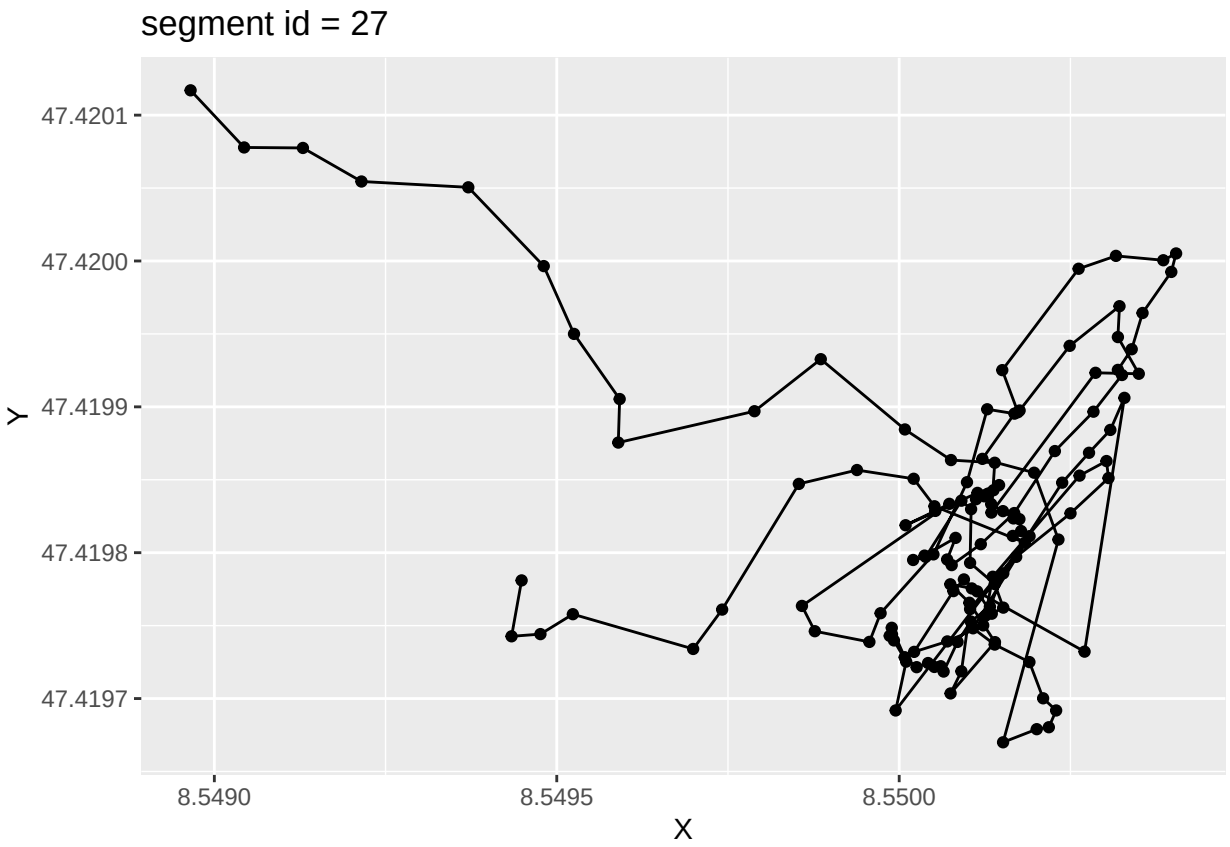
```
## Plot the segments which has a duration more than 5 minutes
ggplot(data = move1f_f1_5min, aes(X, Y, color = segment_id)) +
    geom_path() +
    geom_point() +
    # coord_equal() +
    theme(legend.position = "right")
```



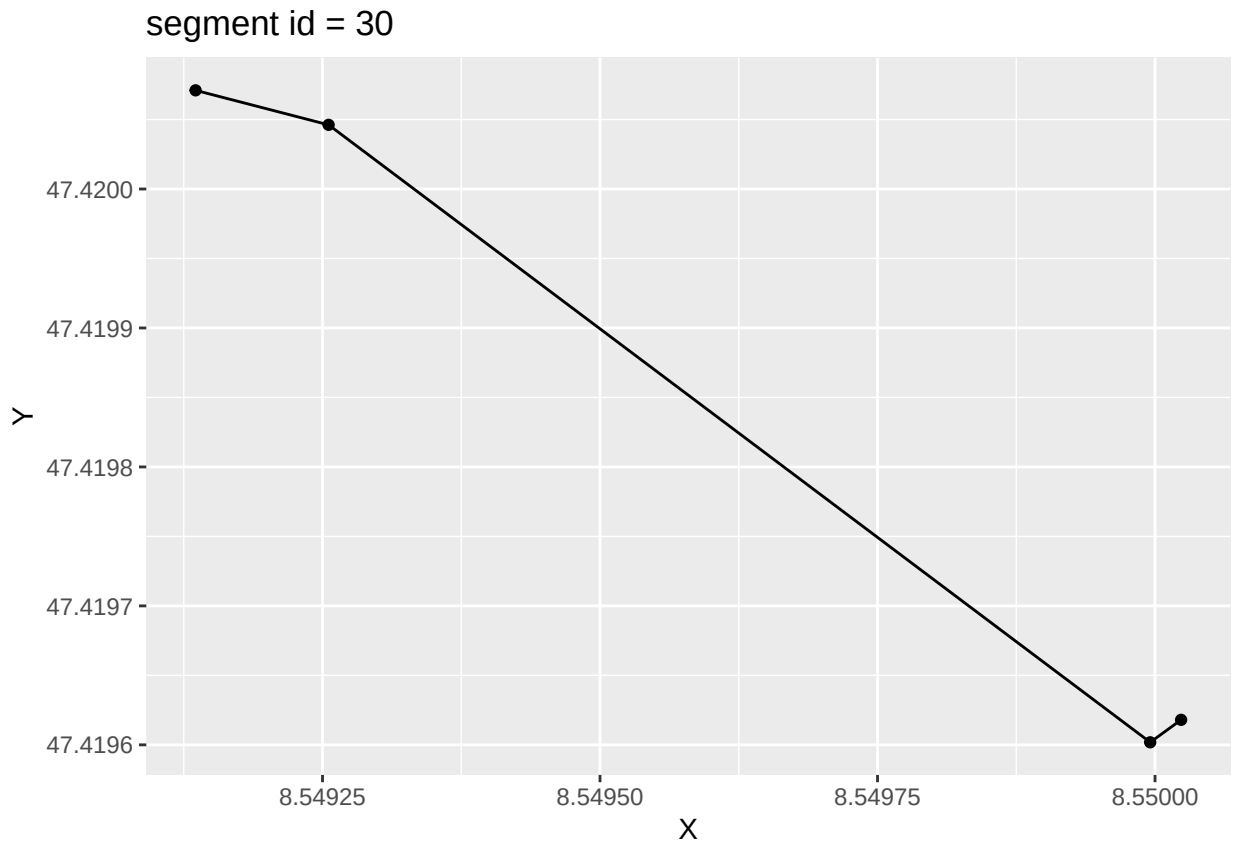

```
## Plot each segment
idlist <- unique(move1f_f1_5min$segment_id)
move1f_f1_5min_1 <- move1f_f1_5min[move1f_f1_5min$segment_id == idlist[1],]
ggplot(data = move1f_f1_5min_1, aes(X, Y)) +
    geom_path() +
    geom_point() +
    ggtitle("segment id = 11")
```



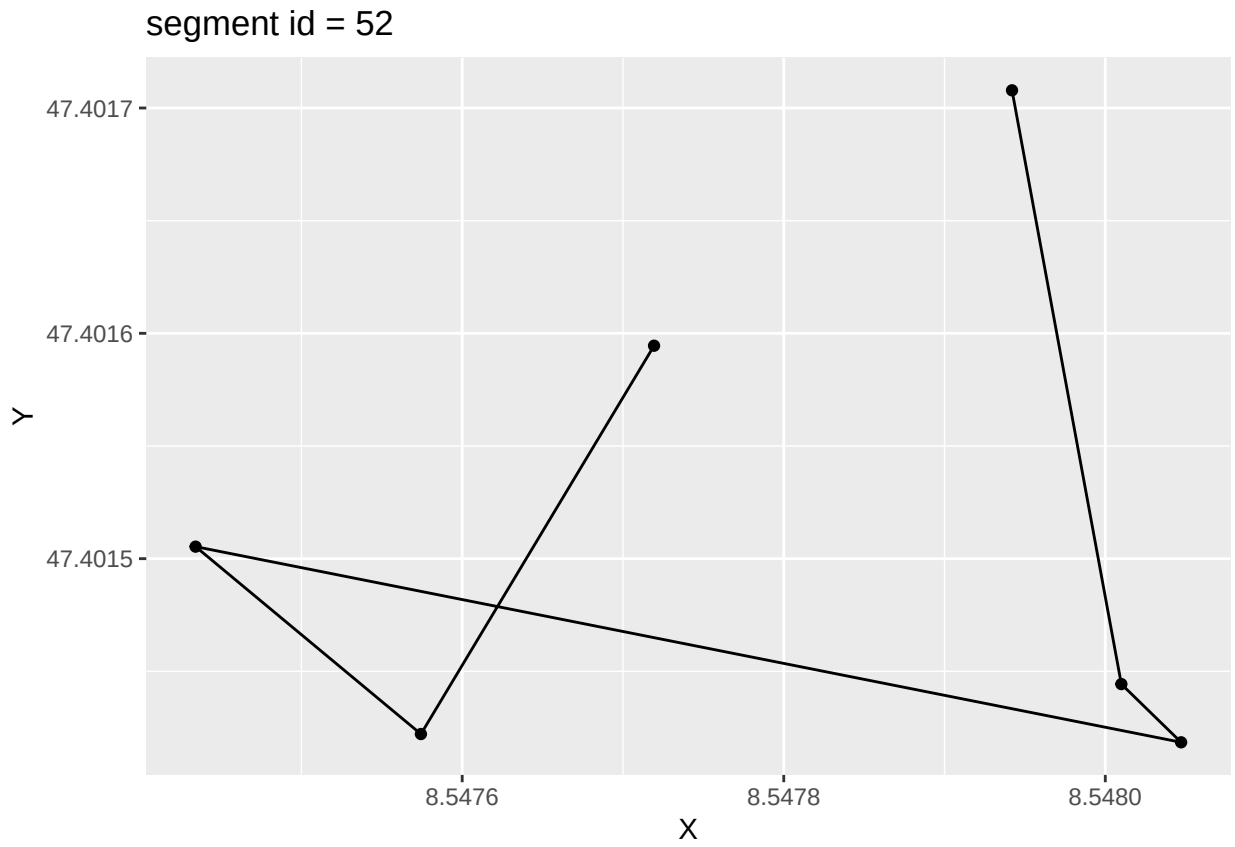

```
move1f_f1_5min_2 <- move1f_f1_5min[move1f_f1_5min$segment_id == idlist[2],]
ggplot(data = move1f_f1_5min_2, aes(X, Y)) +
    geom_path() +
    geom_point() +
    ggtitle("segment id = 27")
```



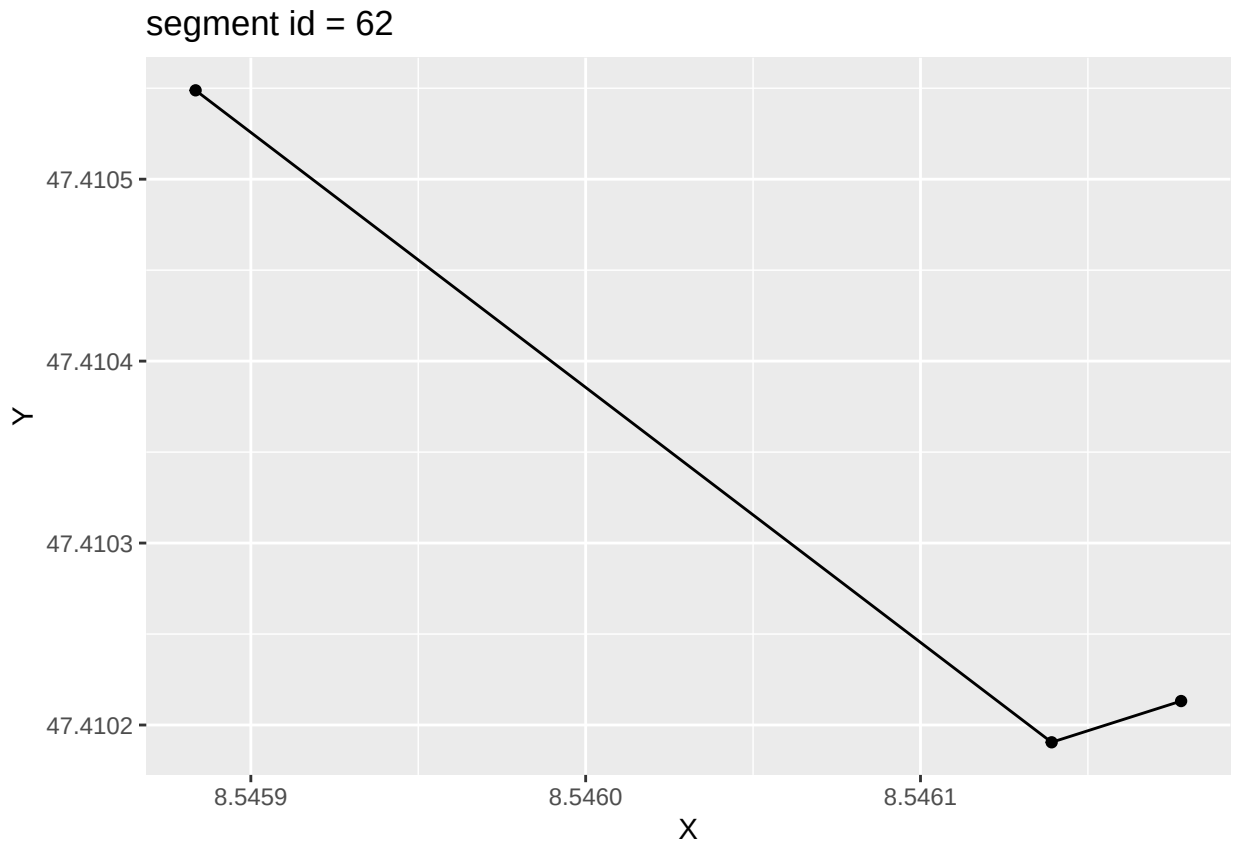

```
move1f_f1_5min_3 <- move1f_f1_5min[move1f_f1_5min$segment_id == idlist[3],]
ggplot(data = move1f_f1_5min_3, aes(X, Y)) +
    geom_path() +
    geom_point() +
    ggtitle("segment id = 30")
```



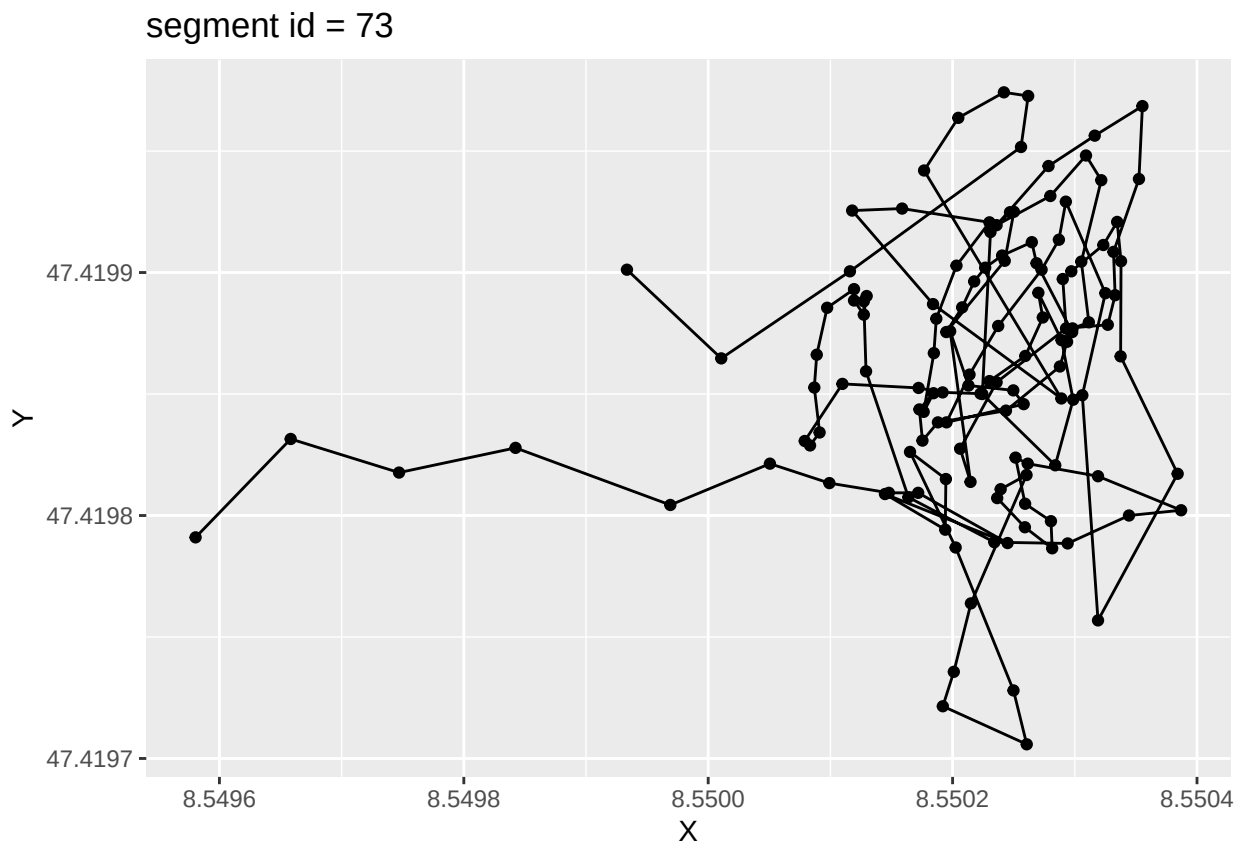

```
move1f_f1_5min_4 <- move1f_f1_5min[move1f_f1_5min$segment_id == idlist[4],]
ggplot(data = move1f_f1_5min_4, aes(X, Y)) +
    geom_path() +
    geom_point() +
    ggtitle("segment id = 52")
```




```
move1f_f1_5min_5 <- move1f_f1_5min[move1f_f1_5min$segment_id == idlist[5],]
ggplot(data = move1f_f1_5min_5, aes(X, Y)) +
    geom_path() +
    geom_point() +
    ggtitle("segment id = 62")
```

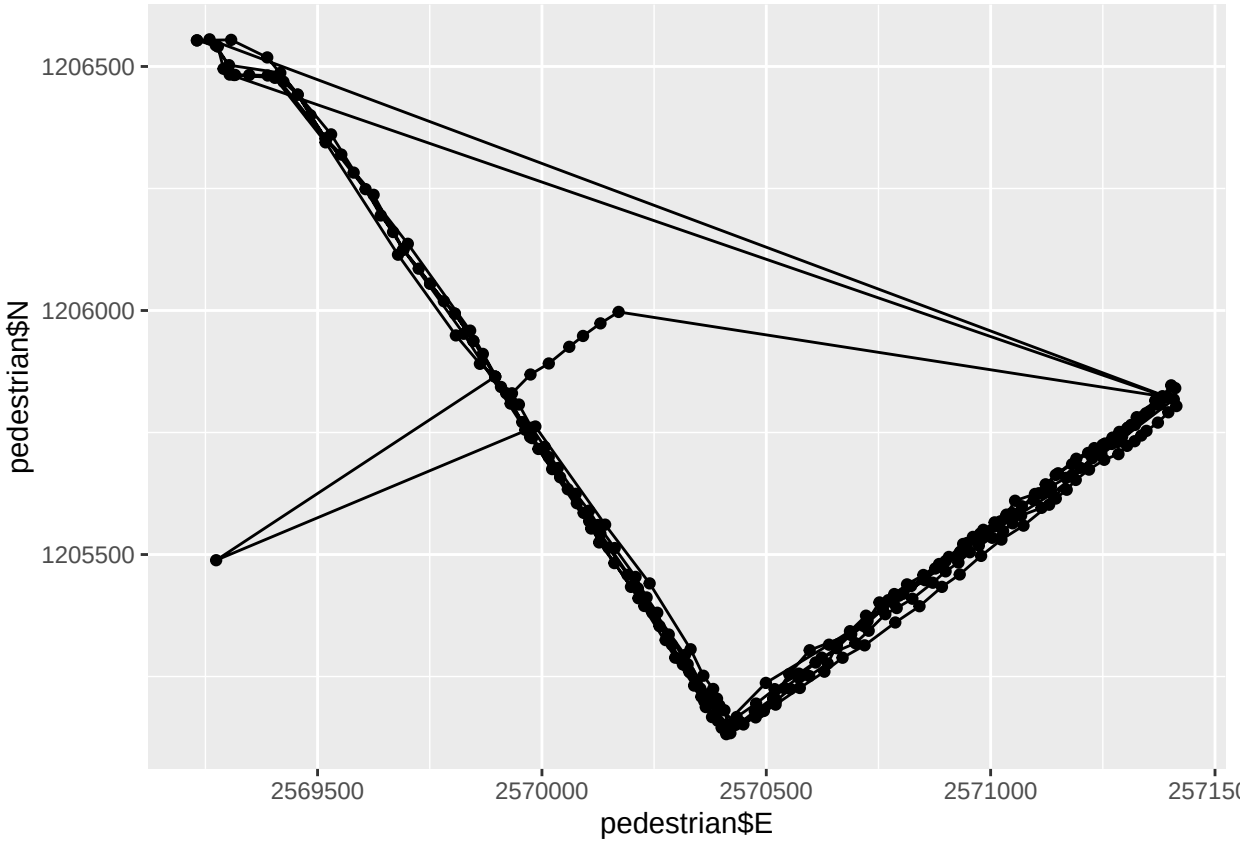



```
move1f_f1_5min_6 <- move1f_f1_5min[move1f_f1_5min$segment_id == idlist[6],]
ggplot(data = move1f_f1_5min_6, aes(X, Y)) +
    geom_path() +
    geom_point() +
    ggtitle("segment id = 73")
```



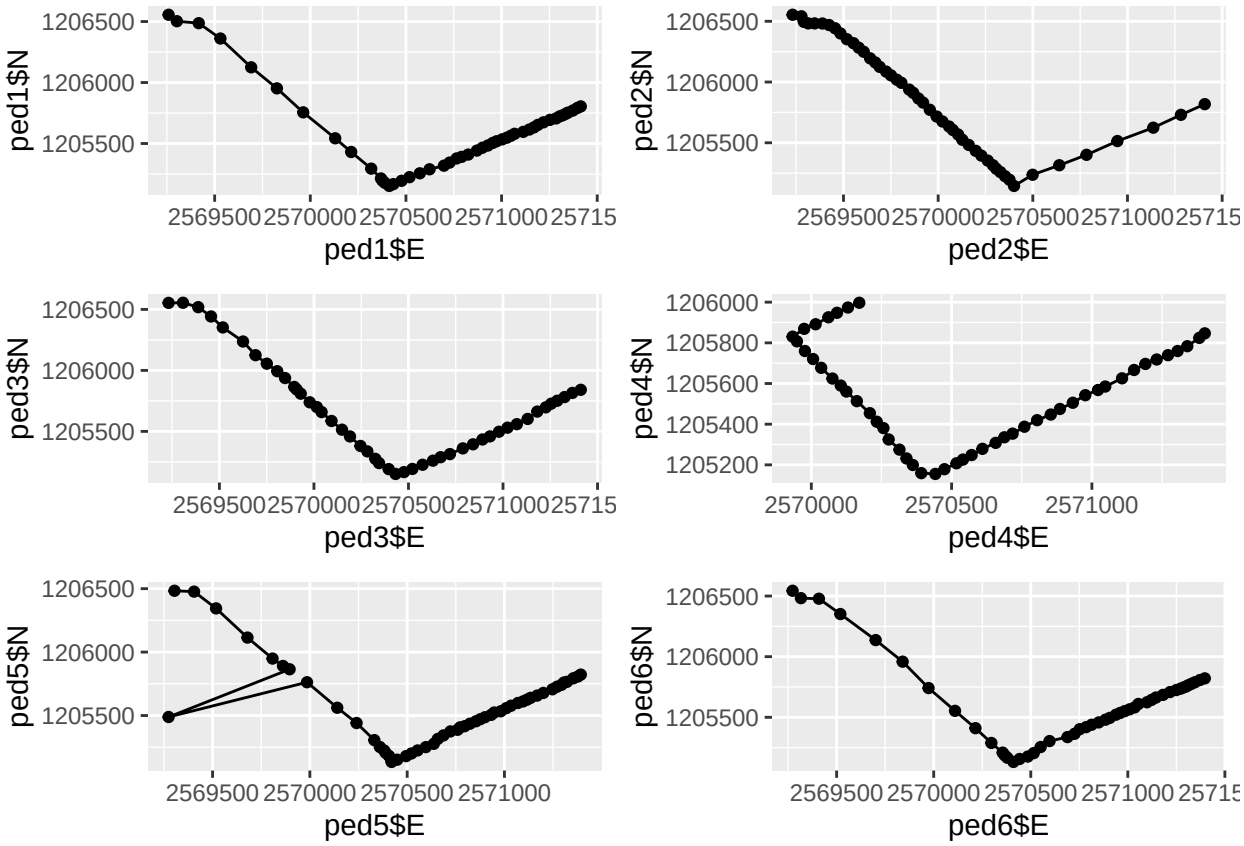
Task 5: Similarity measures

```
## Load the data
pedestrian <- read_delim("pedestrian.csv", ",")
pedestrian <- st_as_sf(pedestrian,
                       coords = c("E", "N"),
                       crs = 2056,
                       remove = FALSE)
ggplot(pedestrian,
       aes(x = pedestrian$E,
           y = pedestrian$N)) +
  geom_point() +
  geom_path()
```




```
ped1 <- pedestrian[pedestrian$TrajID == '1',]
ped2 <- pedestrian[pedestrian$TrajID == '2',]
ped3 <- pedestrian[pedestrian$TrajID == '3',]
ped4 <- pedestrian[pedestrian$TrajID == '4',]
ped5 <- pedestrian[pedestrian$TrajID == '5',]
ped6 <- pedestrian[pedestrian$TrajID == '6',]

pped1 <- ggplot(ped1,
                aes(x = ped1$E, y = ped1$N)) +
  geom_point() +
  geom_path()
pped2 <- ggplot(ped2,
                aes(x = ped2$E, y = ped2$N)) +
  geom_point() +
  geom_path()
pped3 <- ggplot(ped3,
                aes(x = ped3$E, y = ped3$N)) +
  geom_point() +
  geom_path()
pped4 <- ggplot(ped4,
                aes(x = ped4$E, y = ped4$N)) +
  geom_point() +
  geom_path()
pped5 <- ggplot(ped5,
                aes(x = ped5$E, y = ped5$N)) +
  geom_point() +
```

```
  geom_path()
pped6 <- ggplot(ped6,
                aes(x = ped6$E, y = ped6$N)) +
  geom_point() +
  geom_path()

ggarrange(pped1, pped2, pped3,
          pped4, pped5, pped6,
          ncol = 2, nrow = 3)
```



Task 6: Calculate similarity

```
# install.packages("SimilarityMeasures")
# library(SimilarityMeasures)

## Before calculating similarity measures, I think trajectory 1 and 6 are the most similar pairs.

# Convert pedestrian data for each trajectory as matrix
p1 <- data.matrix(data.frame(ped1$E, ped1$N))
p2 <- data.matrix(data.frame(ped2$E, ped2$N))
p3 <- data.matrix(data.frame(ped3$E, ped3$N))
p4 <- data.matrix(data.frame(ped4$E, ped4$N))
p5 <- data.matrix(data.frame(ped5$E, ped5$N))
```

```
p6 <- data.matrix(data.frame(ped6$E, ped6$N))

# Similarity measures: DTW
sm_dtw <- c()
sm_dtw[1] <- DTW(p1, p2, pointSpacing = 1)
sm_dtw[2] <- DTW(p1, p3, pointSpacing = 1)
sm_dtw[3] <- DTW(p1, p4, pointSpacing = 1)
sm_dtw[4] <- DTW(p1, p5, pointSpacing = 1)
sm_dtw[5] <- DTW(p1, p6, pointSpacing = 1)
names <- as.character(c(2:6))
sm_dtw_df <- data.frame(cbind(names, sm_dtw))
sm_dtw_df
```

```
##   names           sm_dtw
## 1     2 37153.3877518943
## 2     3 50785.5113768985
## 3     4               -1
## 4     5               -1
## 5     6 1152.71842265189
```

```
sm_dtw_df$sm_dtw <- round(as.numeric(sm_dtw_df$sm_dtw), digits = 2)
p_dtw <- ggplot(sm_dtw_df, aes(x=names, y = sm_dtw, fill = names)) +
  geom_bar(stat = "identity") +
  ggtitle("DTW") +
  theme(plot.title = element_text(hjust = 0.5)) +
  theme(legend.position = "none") +
  scale_fill_manual(values = c("2" = "coral",
                               "3" = "lavender",
                               "4" = "skyblue",
                               "5" = "khaki",
                               "6" = "seagreen3"))
# Similarity measures: EditDist
sm_ed <- c()
sm_ed[1] <- EditDist(p1, p2, pointDistance = 15)
sm_ed[2] <- EditDist(p1, p3, pointDistance = 15)
sm_ed[3] <- EditDist(p1, p4, pointDistance = 15)
sm_ed[4] <- EditDist(p1, p5, pointDistance = 15)
sm_ed[5] <- EditDist(p1, p6, pointDistance = 15)
sm_ed
```

```
## [1] 46 47 48 46 37
```

```
sm_ed_df <- data.frame(cbind(names, sm_ed))
sm_ed_df
```

```
##   names sm_ed
## 1     2    46
## 2     3    47
## 3     4    48
## 4     5    46
## 5     6    37
```

```
p_ed <- ggplot(sm_ed_df, aes(x=names, y = sm_ed, fill = names)) +
  geom_bar(stat = "identity") +
  ggtitle("EditDist") +
  theme(plot.title = element_text(hjust = 0.5)) +
  theme(legend.position = "none") +
  scale_fill_manual(values = c("2" = "coral",
                               "3" = "lavender",
                               "4" = "skyblue",
                               "5" = "khaki",
                               "6" = "seagreen3"))
# Similarity measures: Frechet
sm_fr <- c()
sm_fr[1] <- Frechet(p1, p2, testLeash = -1)
sm_fr[2] <- Frechet(p1, p3, testLeash = -1)
sm_fr[3] <- Frechet(p1, p4, testLeash = -1)
sm_fr[4] <- Frechet(p1, p5, testLeash = -1)
sm_fr[5] <- Frechet(p1, p6, testLeash = -1)
sm_fr
```

```
## [1]   28.54075 2307.84366 1069.22917  717.98159   38.96272
```

```
sm_fr_df <- data.frame(cbind(names, sm_fr))
sm_fr_df
```

```
##   names            sm_fr
## 1     2 28.5407532415004
## 2     3 2307.84365952892
## 3     4  1069.2291717753
## 4     5 717.981587676952
## 5     6  38.962719035697
```

```
sm_fr_df$sm_fr <- round(as.numeric(sm_fr_df$sm_fr), digits = 2)
p_fr <- ggplot(sm_fr_df, aes(x=names, y = sm_fr, fill = names)) +
  geom_bar(stat = "identity") +
  ggtitle("Frechet") +
  theme(plot.title = element_text(hjust = 0.5)) +
  theme(legend.position = "none") +
  scale_fill_manual(values = c("2" = "coral",
                               "3" = "lavender",
                               "4" = "skyblue",
                               "5" = "khaki",
                               "6" = "seagreen3"))
# Similarity measures: LCSS
sm_lc <- c()
sm_lc[1] <- LCSS(p1, p2, pointSpacing = 1,
                 pointDistance = 20,
                 errorMarg = 2,
                 returnTrans = FALSE)
sm_lc[2] <- LCSS(p1, p3, pointSpacing = 1,
                 pointDistance = 20,
                 errorMarg = 2,
                 returnTrans = FALSE)
```

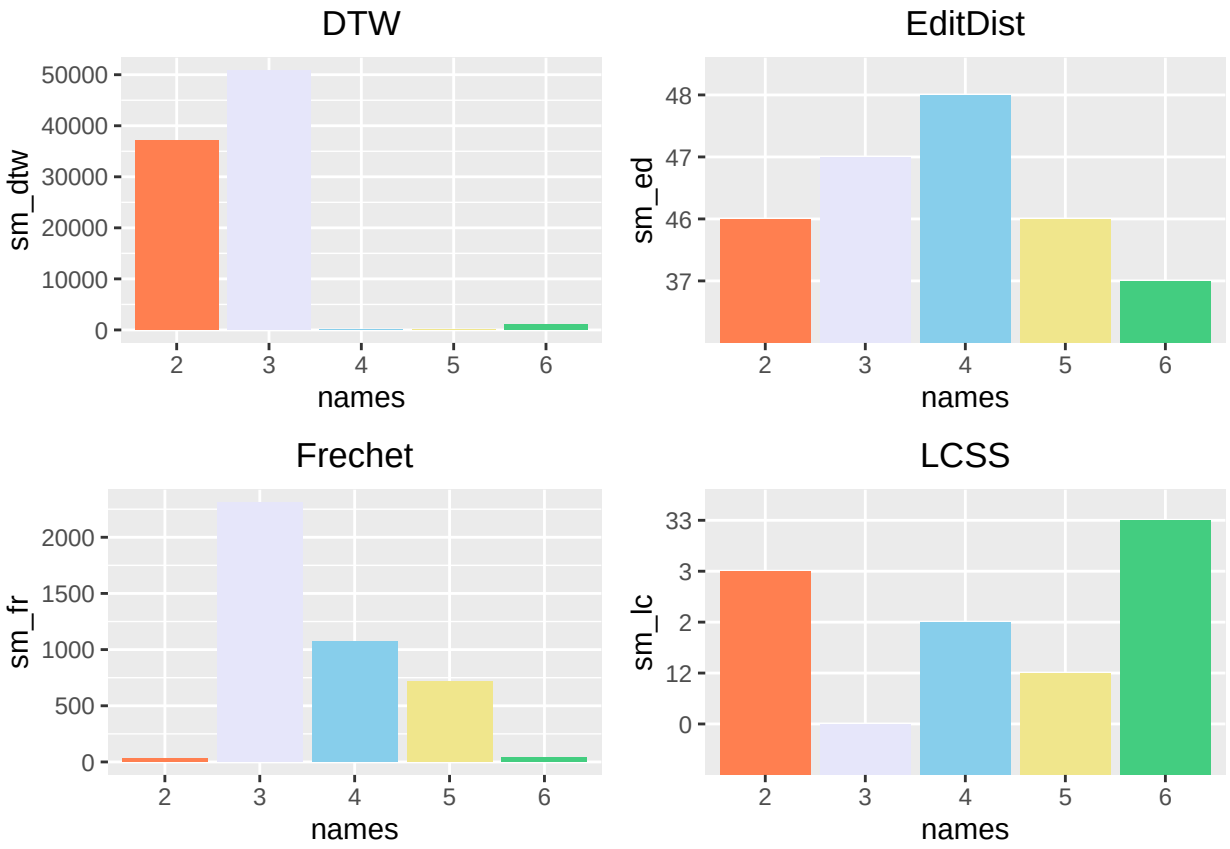
```
sm_lc[3] <- LCSS(p1, p4, pointSpacing = 1,
                 pointDistance = 20,
                 errorMarg = 2,
                 returnTrans = FALSE)
sm_lc[4] <- LCSS(p1, p5, pointSpacing = 1,
                 pointDistance = 20,
                 errorMarg = 2,
                 returnTrans = FALSE)
sm_lc[5] <- LCSS(p1, p6, pointSpacing = 1,
                 pointDistance = 20,
                 errorMarg = 2,
                 returnTrans = FALSE)
sm_lc
```

```
## [1]  3  0  2 12 33
```

```
sm_lc_df <- data.frame(cbind(names, sm_lc))
sm_lc_df
```

```
##   names sm_lc
## 1     2     3
## 2     3     0
## 3     4     2
## 4     5    12
## 5     6    33
```

```
p_lc <- ggplot(sm_lc_df, aes(x=names, y = sm_lc, fill = names)) +
  geom_bar(stat = "identity") +
  ggtitle("LCSS") +
  theme(plot.title = element_text(hjust = 0.5)) +
  theme(legend.position = "none") +
  scale_fill_manual(values = c("2" = "coral",
                               "3" = "lavender",
                               "4" = "skyblue",
                               "5" = "khaki",
                               "6" = "seagreen3"))
ggarrange(p_dtw, p_ed, p_fr, p_lc,
          ncol = 2, nrow = 2)
```


DTW
sm_dtw
50000
40000
30000
20000
10000
0
2
3
4
5
6
names
EditDist
sm_ed
48
47
46
37
2
3
4
5
6
names
Frechet
sm_fr
2000
1500
1000
500
0
2
3
4
5
6
names
LCSS
sm_lc
33
3
2
12
0
2
3
4
5
6
names