Imputing jumps in GPS location

Don Li

08/06/2020

What are we doing?

```
load( "../../dataset/all_SNG.RData" )
source( "utility.R" )
```

If we look at the speeds, we can often find trips where there is `-1` speed between trips.

```
# Number of trips with invalid speed
trips_with_bad_pings = all_data[ , any( speed < 0 ), by = "trj_id" ]
trips_with_bad_pings[ , sum(V1) ]
```

```
## [1] 10092
```

We can look at the distance associated with these negative speeds. We can see that the steps with negative speed tend to be higher on average than steps with positive speed. Clearly, this is wrong, and these negative speeds represent some kind of error. Note that 0 is excluded because 0 is valid on iOS but not Android.

```
all_data[ speed < 0, summary( H_dist ) ]
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##  0.00000  0.01358  0.01911  0.03252  0.02363 20.29763
```

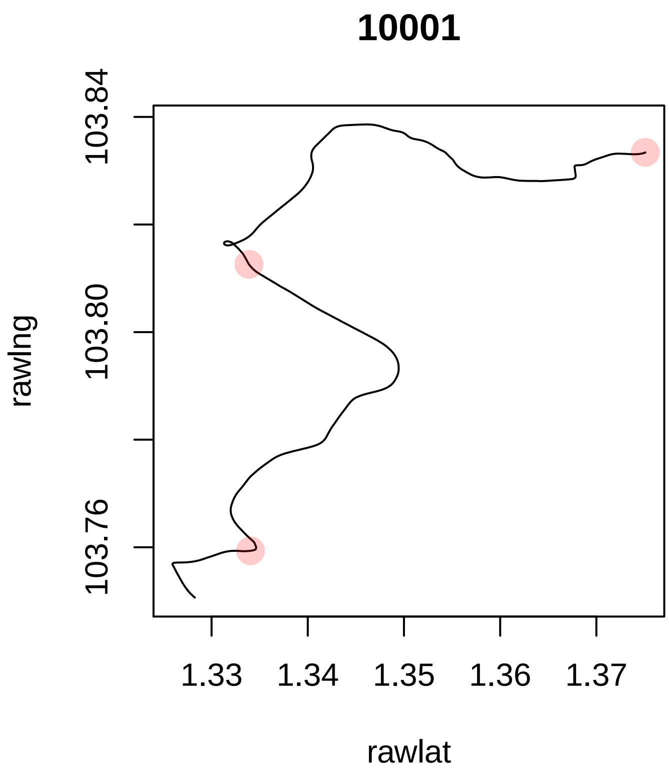
```
all_data[ speed > 0, summary( H_dist ) ]
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##  0.00000  0.01276  0.01841  0.01801  0.02224 18.19138
```

Visualise

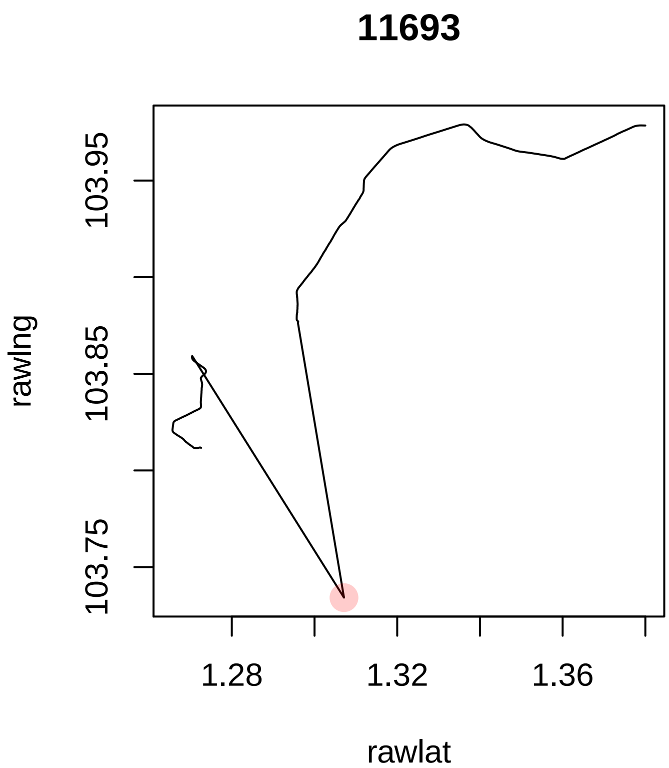
Sometimes the negative speed is a benign thing, such as in the trip below. The red points are where the speed was less than 1.

```
par( pty = "s" )
col = rgb( 1, 0, 0, 0.2 )
bad_trips = trips_with_bad_pings[ V1 == TRUE ]
all_data[ trj_id == 10001,{
    plot( rawlat, rawlng, type = "l",
        main = trj_id[1] )
    points( rawlat[ speed < 0 ], rawlng[ speed < 0 ],
        cex = 2, pch = 16, col = col   )
} ]
```




```
## NULL
```

```r
par( pty = "s" )
all_data[ trj_id == 11693,{
    plot( rawlat, rawlng, type = "l",
        main = trj_id[1] )
    points( rawlat[ speed < 0 ], rawlng[ speed < 0 ],
        cex = 2, pch = 16, col = col )
} ]
```




```
## NULL
```

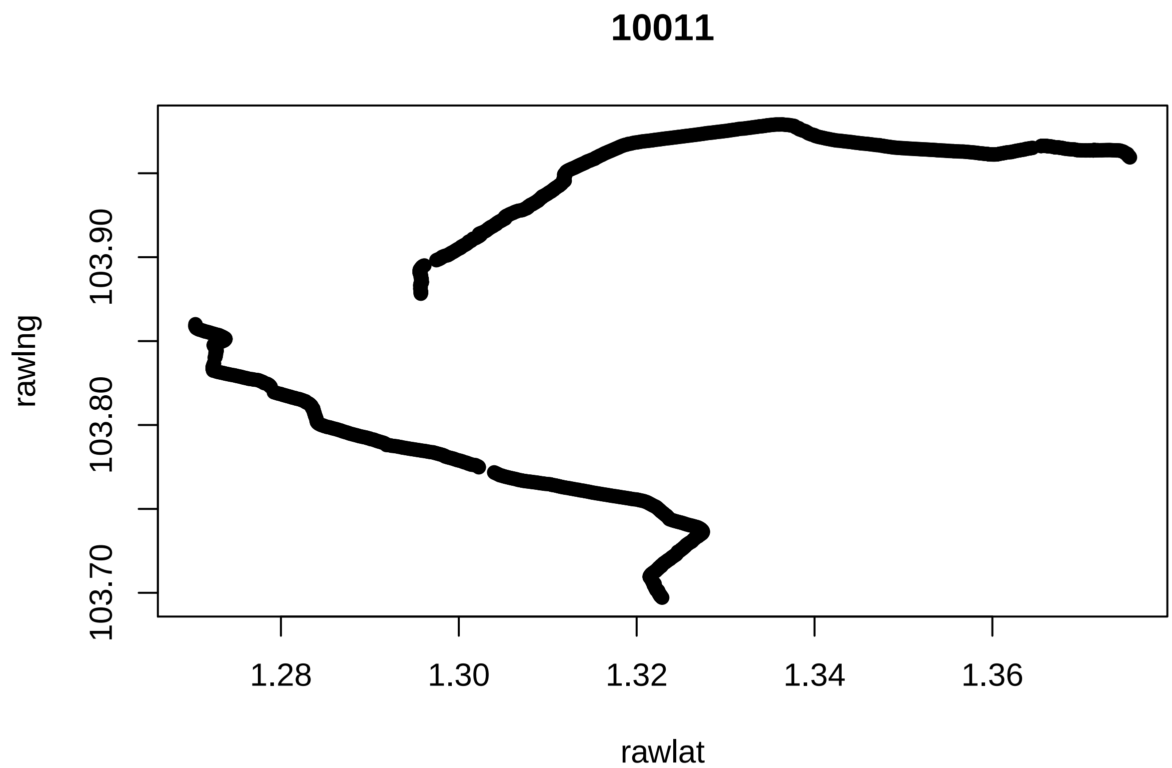
Conclusion

I do not think the negative speed in itself is worrying. But it could be symptomatic of something worse, like a jump in the GPS pings.

Analyse trips with big jumps

An example:

```
all_data[ trj_id == 10011, {
    plot(rawlat, rawlng, pch = 16, main = trj_id[1])
    } ]
```




```
## NULL
```

This one seems fine. It looks like their GPS stopped working for a bit, or they went into a tunnel or something.

So, I think the cases where we might worry is when there are two points with very large distances. In the example plotted previously:

```
all_data[ trj_id == 11693 ][ which(speed < 0) + -2:2 ]
```

```
##    trj_id  osname   rawlat   rawlng     speed               date_ weekday hour
## 1:  11693 android 1.270380 103.8591 18.667429 2019-04-18 22:43:11     Thu   22
## 2:  11693 android 1.270421 103.8592 18.687794 2019-04-18 22:43:12     Thu   22
## 3:  11693 android 1.307128 103.7342 -1.000000 2019-04-18 22:44:28     Thu   22
## 4:  11693 android 1.295993 103.8763  3.303370 2019-04-18 22:50:43     Thu   22
## 5:  11693 android 1.295996 103.8764  4.502514 2019-04-18 22:50:44     Thu   22
##    is_weekend rush_hour       H_dist     H_dist2 time_diff
## 1:      FALSE        No  0.018720461  0.01866304         1
## 2:      FALSE        No  0.018390294  0.01782980         1
## 3:      FALSE        No 14.501549257 13.95153667        76
## 4:      FALSE        No 15.867322948 15.82567291       375
## 5:      FALSE        No  0.003442773  0.00342823         1
```

An imputed journey for 11693 is shown below. We impute by finding points where there are two steps larger than 5 in a row. The black line is the original, the red is the one with the imputed point.

```
test_data = all_data[ trj_id == 11693 ]
two_window_threshold = function( x, threshold ){
    (x > threshold) & (c( x[-1], F ) > threshold)
}
```

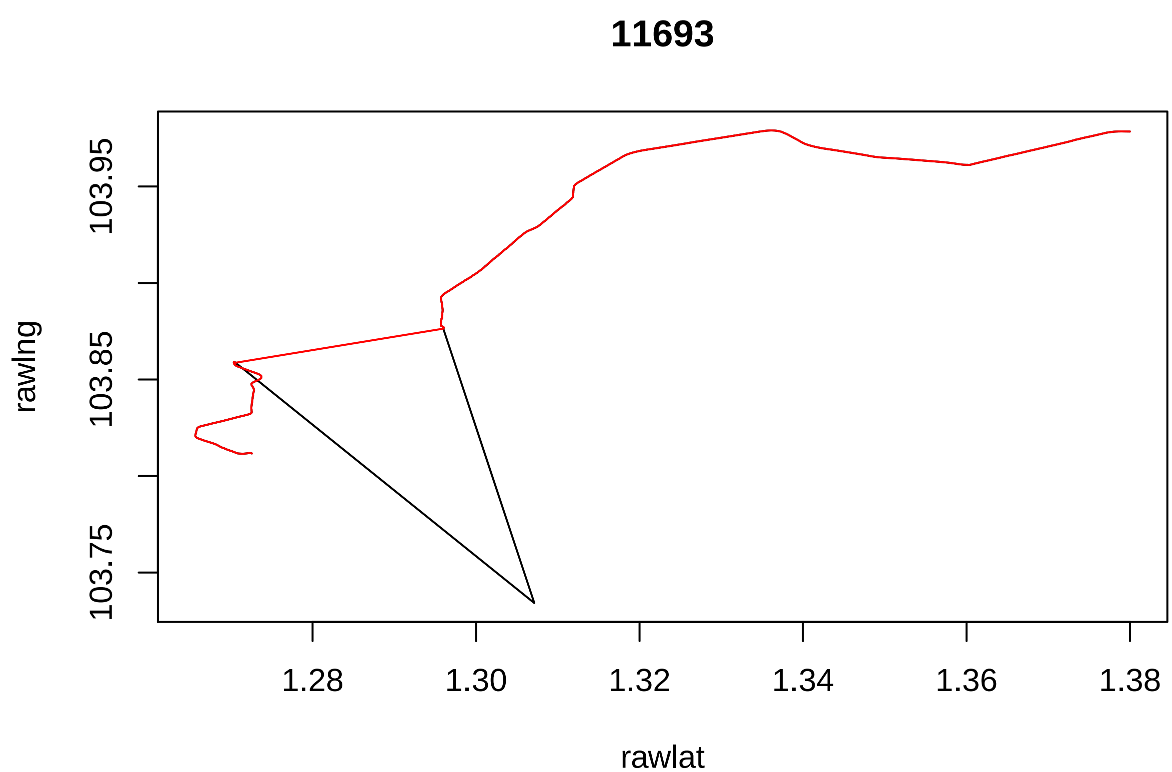
```
test_data[ , c( "rawlat", "rawlng" ) := {
    missing_dist = which( two_window_threshold( H_dist, 5 ) )
    for ( missing in missing_dist ){
        neighbours = missing - 5:5
        around_lat = rawlat[ neighbours ]
        around_lng = rawlng[ neighbours ]
        interpolate_lat = mean( around_lat, na.rm = T )
        interpolate_lng = mean( around_lng, na.rm = T )

        rawlat[ missing ] = interpolate_lat
        rawlng[ missing ] = interpolate_lng
    }
    list( rawlat, rawlng )
    } ]
test_data[ , H_dist := c( 0, haversine(rawlat, rawlng) ) ]

all_data[ trj_id == 11693,{
    plot( rawlat, rawlng, type = "l",
        main = trj_id[1])
    } ]
```

```
## NULL
```

```
test_data[ , {
    lines( rawlat, rawlng, type = "l", col = "red" )
} ]
```



```
## NULL
```

And the distances seem more reasonable with the imputed values.

```r
# Crow flies distance
all_data[ trj_id == 11693, haversine( rawlat[c(1,.N)], rawlng[c(1,.N)] ) ]
```

```
## [1] 22.08126
```

```r
# Path distance
all_data[ trj_id == 11693, sum( H_dist, na.rm = T ) ]
```

```
## [1] 55.84373
```

```r
# Path with imputation
test_data[ , sum(H_dist) ]
```

```
## [1] 29.02173
```

Do this with the rest of the data:

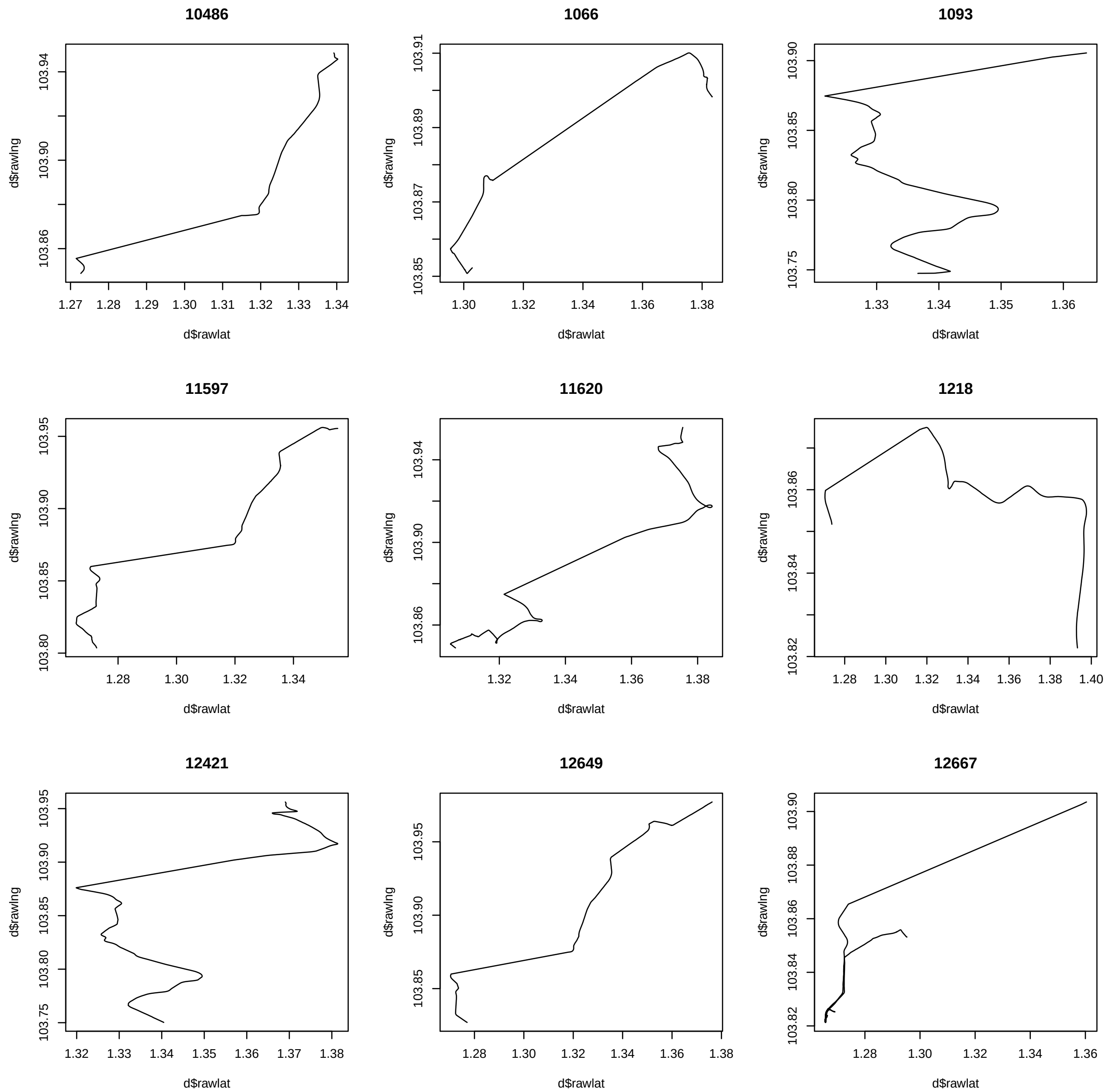
```r
all_data[ , c( "rawlat", "rawlng" ) := {
    missing_dist = which( two_window_threshold( H_dist, 5 ) )
    for ( missing in missing_dist ){
        neighbours = missing - 5:5
        around_lat = rawlat[ neighbours ]
        around_lng = rawlng[ neighbours ]
        interpolate_lat = mean( around_lat, na.rm = T )
        interpolate_lng = mean( around_lng, na.rm = T )

        rawlat[ missing ] = interpolate_lat
        rawlng[ missing ] = interpolate_lng
    }
    list( rawlat, rawlng )
}, by = "trj_id" ]

all_data[ , c("H_dist", "H_dist2") := {
    h_dist = haversine( rawlat, rawlng )
    h1 = c( 0, h_dist  )
    h_dist2 = haversine( rawlng, rawlat )
    h2 = c( 0, h_dist2 )
    list( h1, h2 )
}, by = "trj_id" ]
```

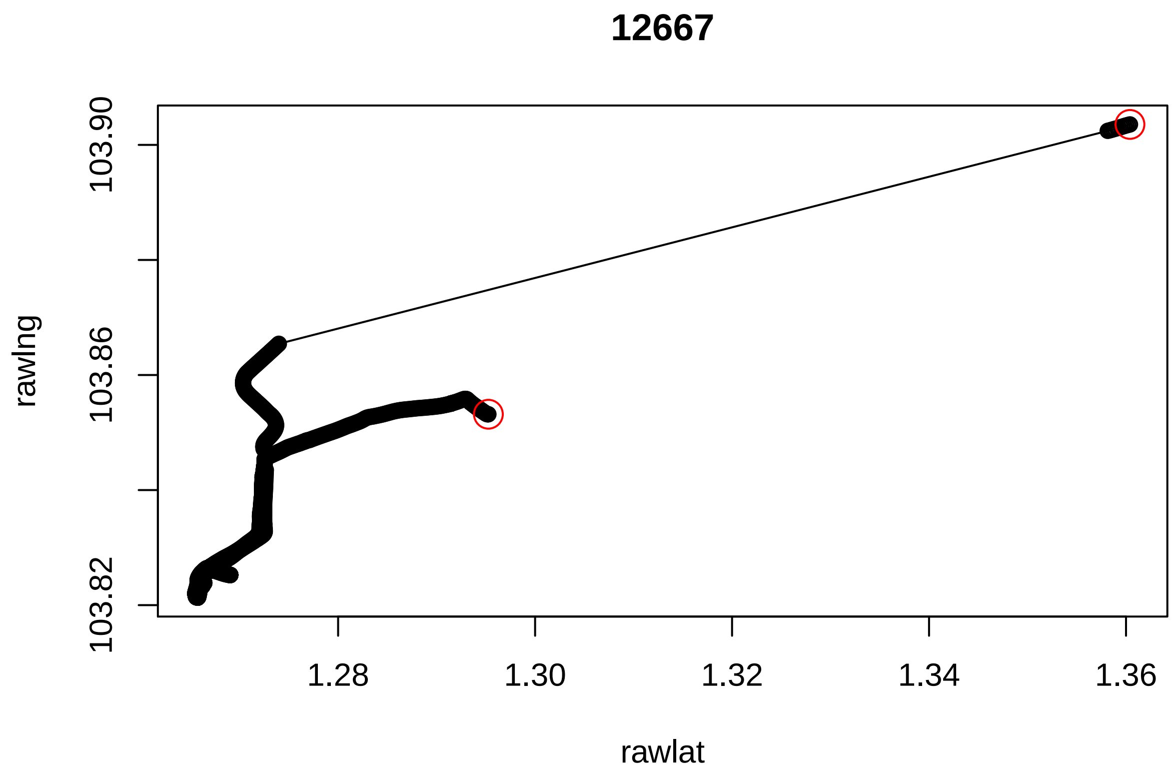
Try to find some trips with big jumps.

```r
par( mfrow= c(3,3))
bad_trips = all_data[ H_dist > 5, unique(trj_id) ]
for ( trj in bad_trips[1:9] ){
    d = all_data[ trj_id == trj ]
    plot( d$rawlat, d$rawlng, type = "l",
        main = d$trj_id[1])
}
```



Examnine 12667. Seems okay. GPS probably just broke at the start.

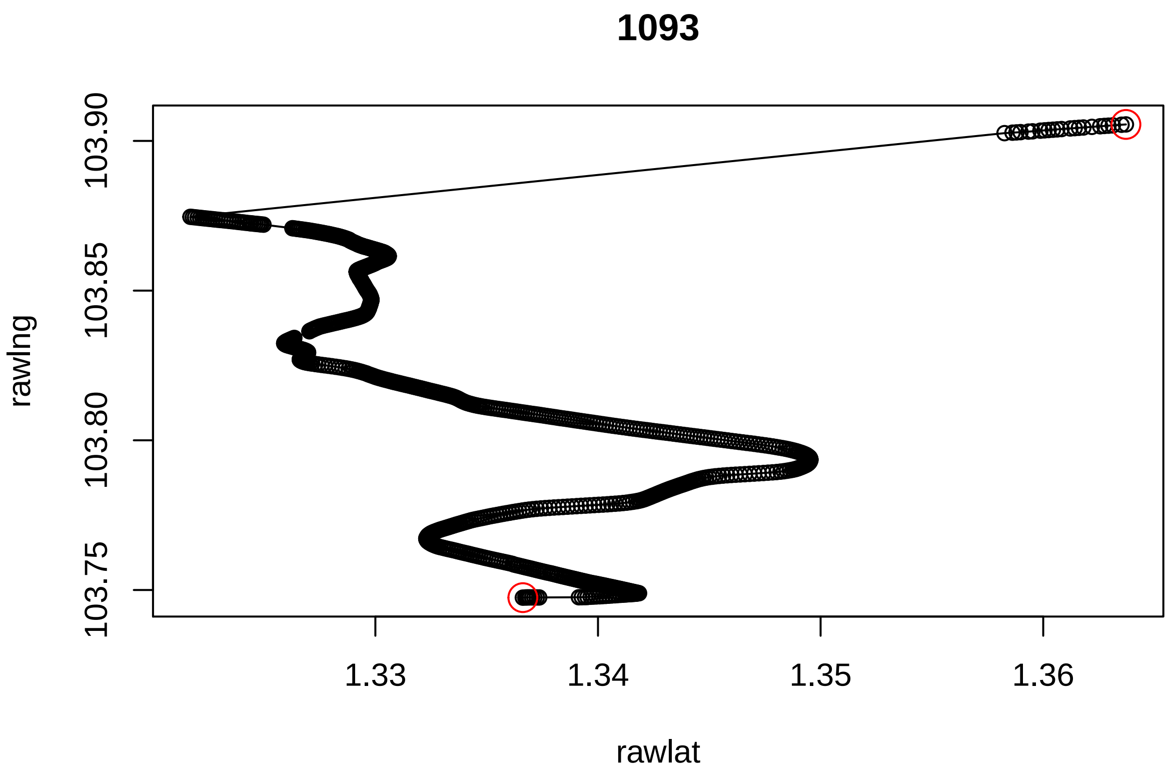
```r
par( mfrow = c(1,1 ))
trip = 12667
all_data[ trj_id == trip, {
    plot( rawlat, rawlng, main = trj_id[1], type = "o" )
    points( rawlat[c(1,.N)], rawlng[c(1,.N)], col = "red", cex = 2 )
} ]
```




```
## NULL
```

Examnine 1093 Seems okay. GPS probably just broke at the start.

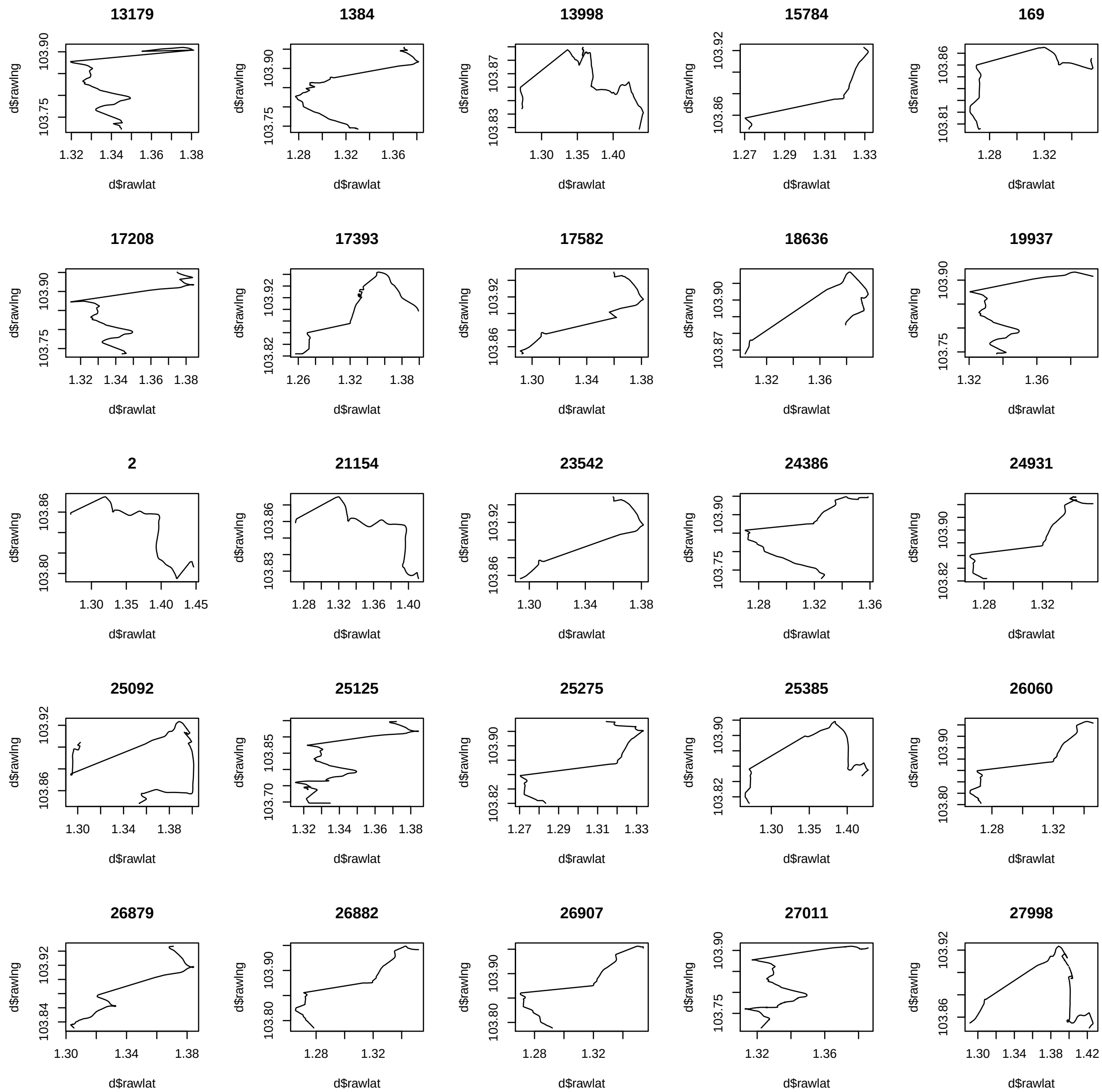
```
par( mfrow = c(1,1 ))
trip = 1093
all_data[ trj_id == trip, {
    plot( rawlat, rawlng, main = trj_id[1], type = "o" )
    points( rawlat[c(1,.N)], rawlng[c(1,.N)], col = "red", cex = 2 )
} ]
```




```
## NULL
```

Try again:

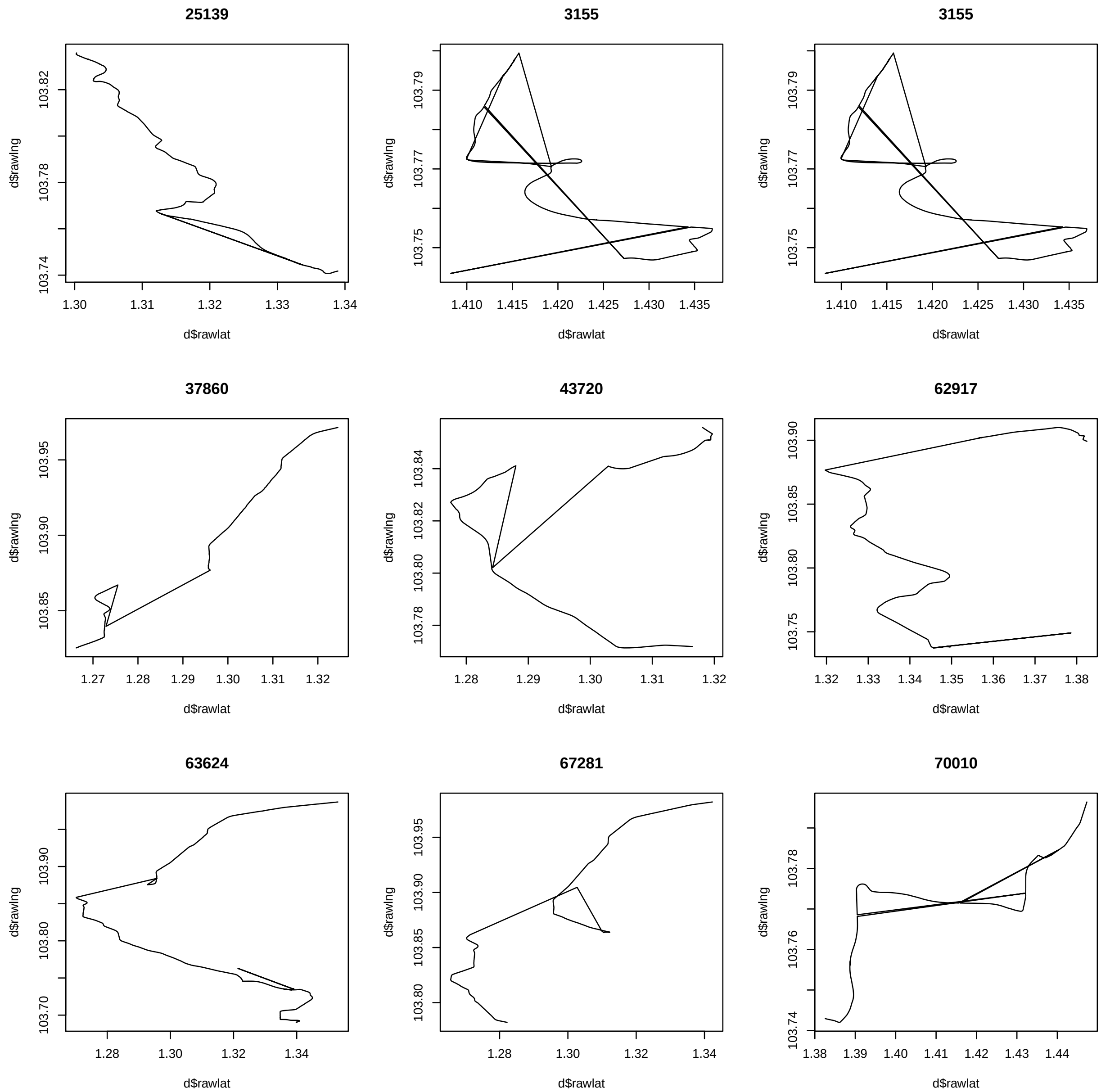
```
par( mfrow= c(5,5))
bad_trips = all_data[ H_dist > 5, unique(trj_id) ]
for ( trj in bad_trips[1:25 + 9] ){
    d = all_data[ trj_id == trj ]
    plot( d$rawlat, d$rawlng, type = "l",
        main = d$trj_id[1])
}
```



Seems okay.

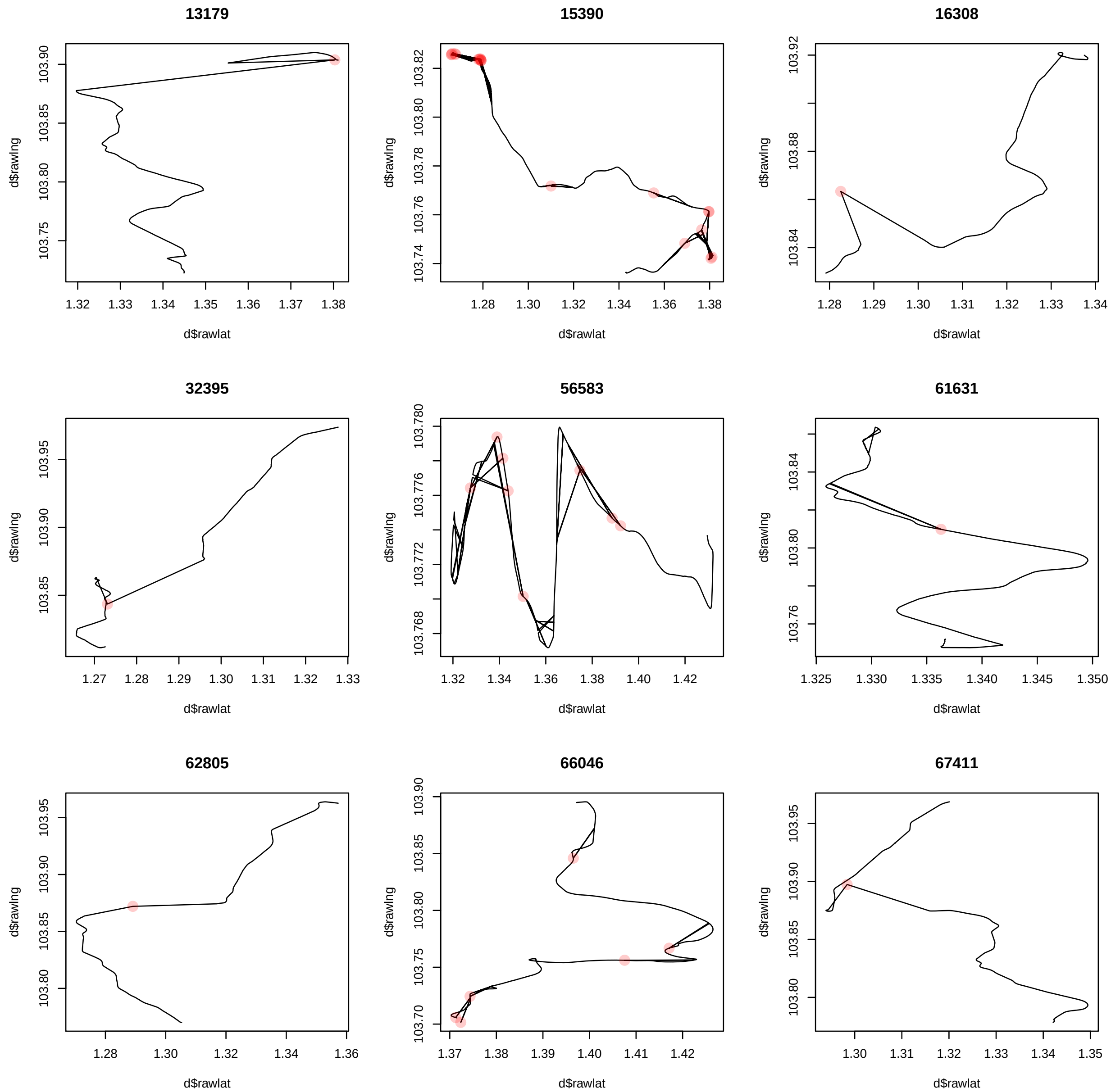
Round 2

Try detecting bad jumps with a smaller threshold



Set new threshold for jumps to 3.

Round 3



Went up to two and that seemed good enough. Still a couple of weird ones, but a pretty small minority.

Conclusions:

For cleaning the data, we will interpolate the coordinates of points where that point and the next point have a large distance from the previous point. This represents the situation where there is one stray ping somewhere.

The distance we will use is 2km (Haversine).