Nearby roadways

Danielle Medgyesi

12/20/2022

OpenStreetMap

Download all street types from osm for study region

https://wiki.openstreetmap.org/wiki/Key:highway#Roads

- **trunk**=The most important roads in a country's system that aren't motorways. (Need not necessarily be a divided highway.)
- **primary**= The next most important roads in a country's system. (Often link larger towns.)
- **secondary**= The next most important roads in a country's system. (Often link towns.)
- **tertiary**= The next most important roads in a country's system. (Often link smaller towns and villages)
- **unclassified**= The least important through roads in a country's system – i.e. minor roads of a lower classification than tertiary, but which serve a purpose other than access to properties. (Often link villages and hamlets.) The word 'unclassified' is a historical artefact of the UK road system and does not mean that the classification is unknown; you can use highway=road for that.
- **residential**= Roads which serve as an access to housing, without function of connecting settlements. Often lined with housing.
- **service**= For access roads to, or within an industrial estate, camp site, business park, car park, alleys, etc. Can be used in conjunction with service=* to indicate the type of usage and with access=* to indicate who can use it and in what circumstances.
- **track**= Roads for mostly agricultural or forestry uses. To describe the quality of a track, see tracktype=*. Note: Although tracks are often rough with unpaved surfaces, this tag is not describing the quality of a road but its use. Consequently, if you want to tag a general use road, use one of the general highway values instead of track.
- **road**= A road/way/street/motorway/etc. of unknown type. It can stand for anything ranging from a footpath to a motorway. This tag should only be used temporarily until the road/way/etc. has been properly surveyed. If you do know the road type, do not use this value, instead use one of the more specific highway=* values. footway= For designated footpaths; i.e., mainly/exclusively for pedestrians. This includes walking tracks and gravel paths. If bicycles are allowed as well, you can indicate this by adding a bicycle=yes tag. Should not be used for paths where the primary or intended usage is unknown. Use highway=pedestrian for pedestrianised roads in shopping or residential areas and highway=track if it is usable by agricultural or similar vehicles.

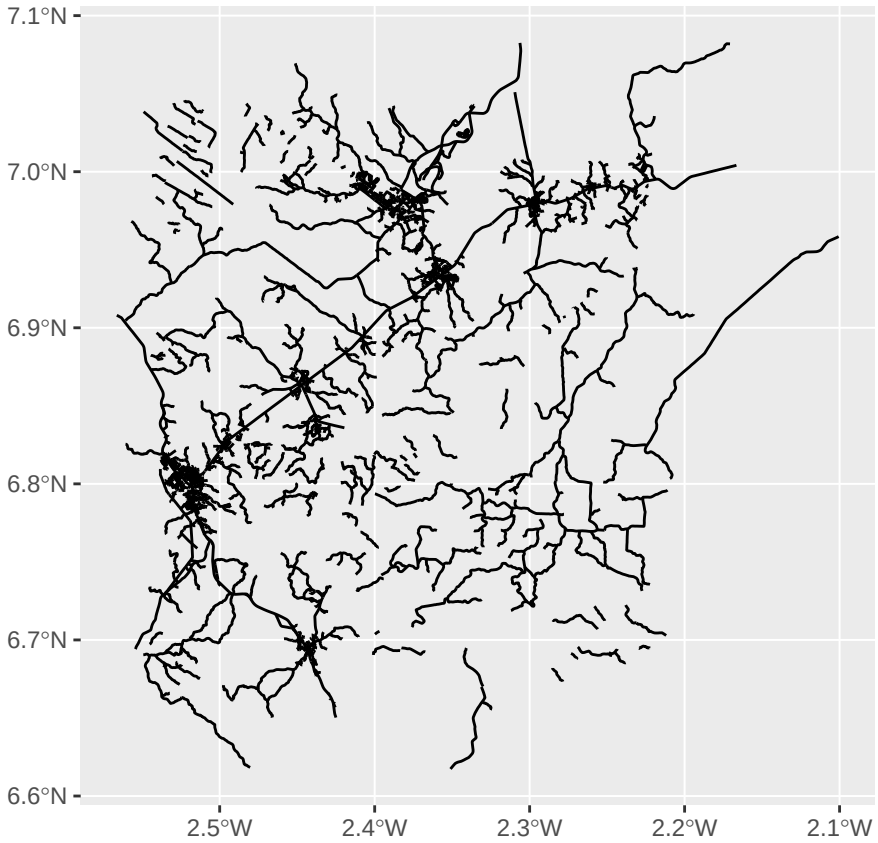
```
#search study region


boundaries <- opq(bbox = 'GH-AF') %>%
  add_osm_feature(key = 'admin_level', value=6) %>%
```

```
  osmdata_sf %>% unique_osmdata

districts<-boundaries$osm_multipolygons

D1<- districts %>% filter(name=="Asutifi South District")

D1<- D1 %>% st_transform(4326)

#first filter buildings inside xy limits
D_box<- st_bbox(D1)

location <- D_box %>% opq()

#quesry all types of roads

all_types<- available_tags("highway")


 Streets <- location %>%
   add_osm_feature(key = "highway",
                   value = all_types) %>%
   osmdata_sf()


Street_lines<- Streets$osm_lines



kable(Street_lines %>% data.frame()%>% group_by(highway) %>%
        summarise(n()),
      caption = "Number of streets in study region")
```

Table 1: Number of streets in study region

highway	n()
footway	57
path	187
primary	1
residential	1578
secondary	9
service	83
track	102
trunk	21
trunk_link	2
unclassified	427

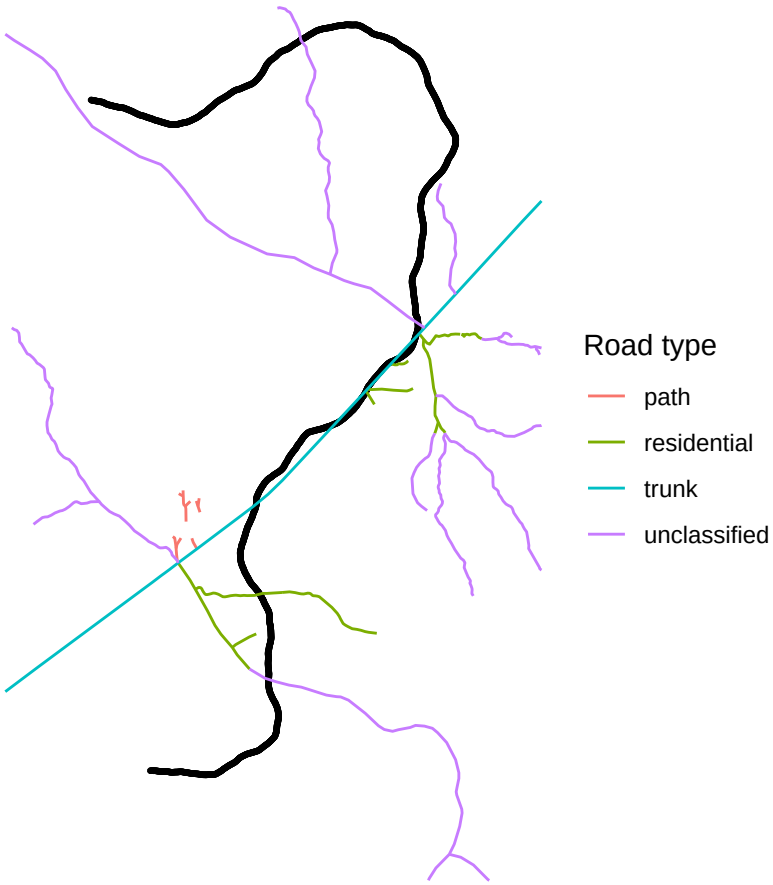
```
ggplot()+
  geom_sf(data=Street_lines)
```



Simulated GPS trajectory

```
# Create a random trajectory

set.seed(333)
trj <- TrajGenerate(n = 2880, stepLength = 2, angularErrorSd = 0.05, random = TRUE)
#trj <- TrajRotate(trj, pi / 1.5, relative = FALSE)
trj<- TrajReverse(trj)
trj<- TrajTranslate(trj, 1000, 2500)

#for illustration, use a community centroid as the home coordinates
Home_lon<- -2.43
Home_lat<- 6.85

trj<- trj %>% mutate(x=Home_lon + (x / 6378000) * (200 / pi) / cos(Home_lon * pi/200),
                     y= Home_lat  + (y / 6378000) * (200 / pi))

trj_sf <- st_as_sf(trj, coords = c("x", "y"),
                crs = 4326)


p<- ggplot()+geom_sf(data=trj_sf)
```

```
y_range<- ggplot_build(p)$layout$panel_params[[1]]$y_range+c(-0.003,0.003)
x_range<- ggplot_build(p)$layout$panel_params[[1]]$x_range+c(-0.003,0.003)


filt_bbox <- sf::st_bbox(c(xmin = x_range[1],
                           ymin = y_range[1],
                           xmax = x_range[2],
                           ymax = y_range[2]),
                         crs = st_crs(4326))%>%
  sf::st_as_sfc(.)



Roadfiltr<-sf::st_intersection(Street_lines, filt_bbox)



ggplot()+
  geom_sf(data=trj_sf, size=0.5)+
  geom_sf(data = Roadfiltr, aes(color=highway))+
  theme_void()+
  labs(color="Road type")
```



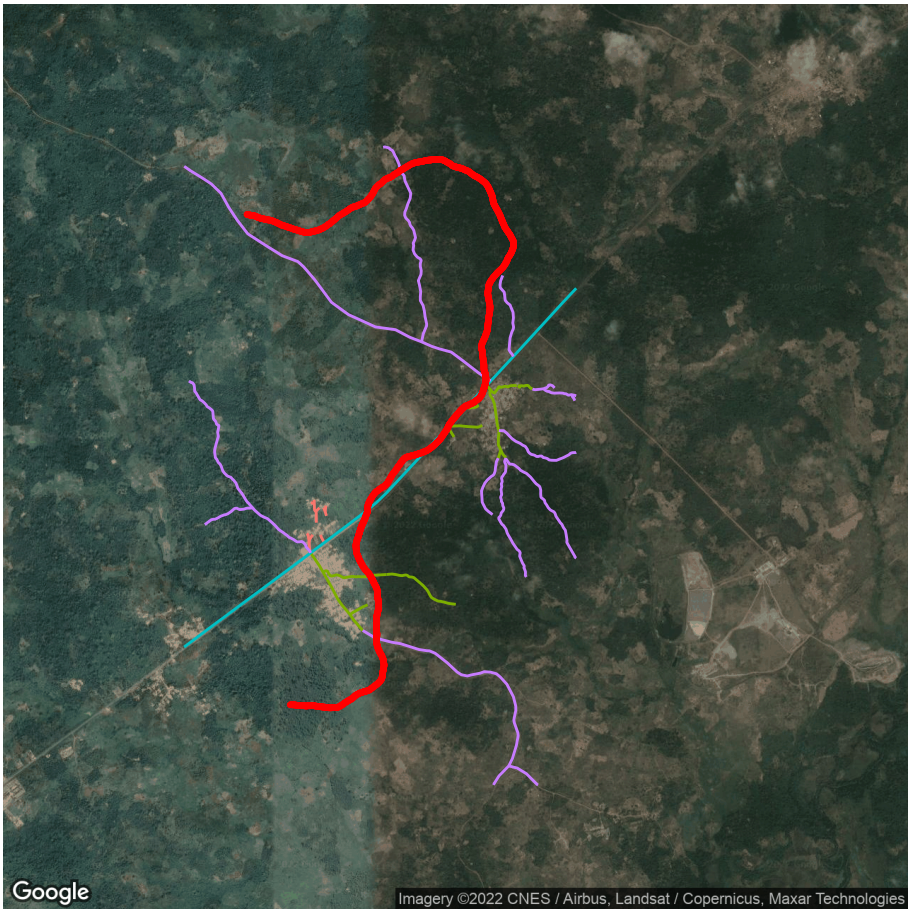
In order to access satellite data from Google, users must obtain a valid Google Maps API key

To do so, you can create an account with Google: https://mapsplatform.google.com/

- Begin a new project and create credentials
- I created an API key, selecting "restrict key" > "Maps JavaScript API"
- API can then be called into R using the "register_google" function
- Make sure your API is secure; charges may be apply to high-volume users

If you do not wish to create an API, skip code below and refer to plain ggplot maps

```
#map_key <- ""
#register_google(key=map_key)

base<- get_map(location = c(-2.41, 6.89), zoom=14, maptype = "satellite")


ggmap(base)+
  coord_sf(crs = st_crs(4326)) +
  geom_sf(data = Roadfiltr, aes(color=highway), size=1.1,
          inherit.aes = FALSE)+
   geom_sf(data = trj_sf,color="red", size=0.5,
          inherit.aes = FALSE)+
  theme(axis.line = element_blank(),
        axis.text = element_blank(),
        axis.ticks = element_blank(),
        plot.margin = unit(c(0, 0, -1, -1), 'lines')) +
  xlab('') +
  ylab('')+
  labs(color="Road type")
```



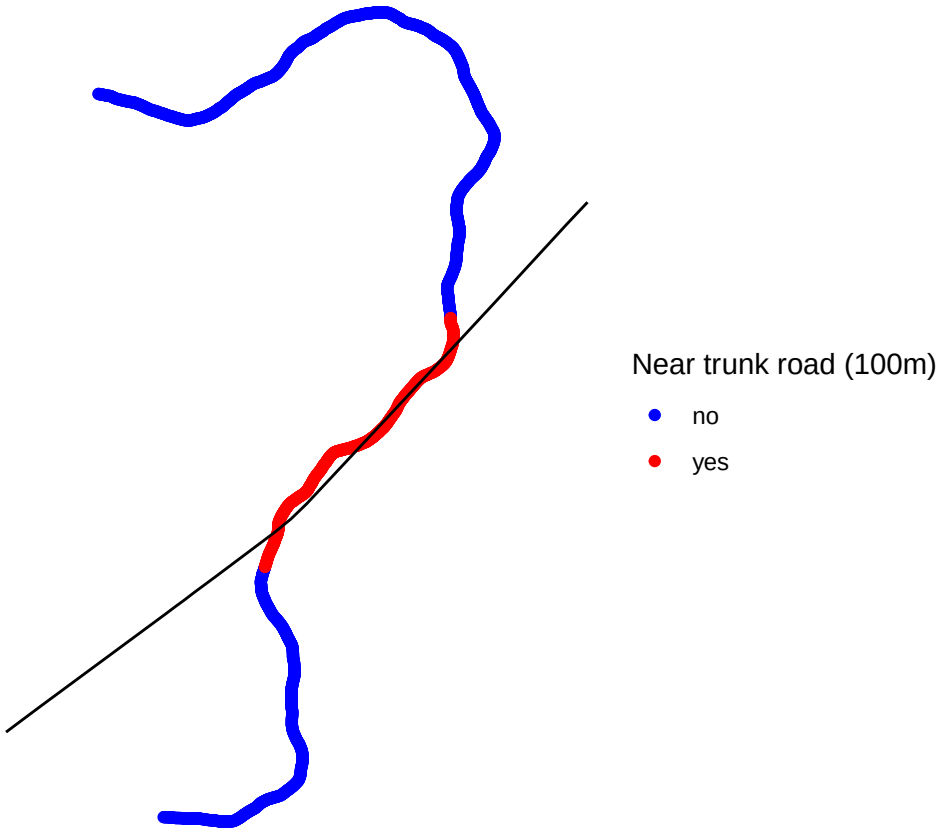
Identify nearby roadways and density (100m buffer)

```
#Transform to Ghana crs
#https://epsg.io/?q=Ghana
Streets_sf<- Roadfiltr %>% st_transform(32630)
trj_sf <- trj_sf %>% st_transform(32630)
#create unique ID for each point
trj_sf<- trj_sf %>% mutate(ID=1:nrow(.))

#Split street dataset into list by street type
Streets_sf_list<- split(Streets_sf, f= Streets_sf$highway)

#Create 100m buffer around GPS points
trj_sf100 <- st_buffer(trj_sf, dist = 100)
```

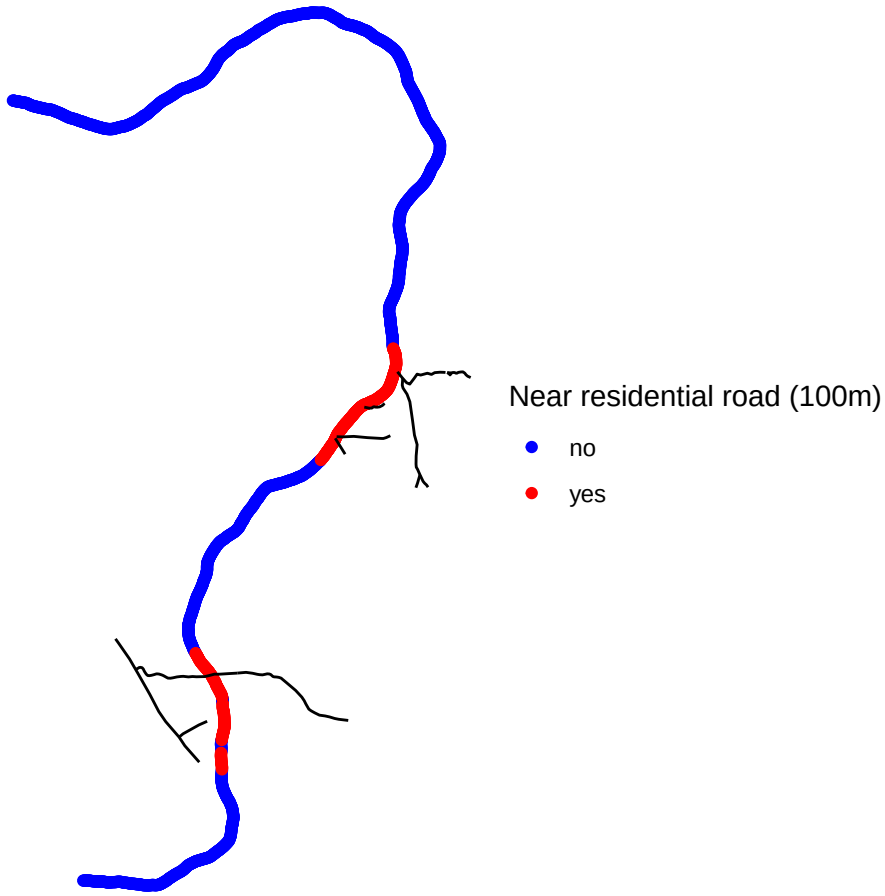
```
#loop through each road type
trj_roadways<- list()
for (i in 1:length(Streets_sf_list)) {
# identify roads that are within 100m
intrsct = st_intersection(trj_sf100, Streets_sf_list[[i]])
# save the length of each road segment
intrsct$len = st_length(intrsct)
# sume road length for each point
sum_len = intrsct %>% group_by(ID) %>% summarise(len = sum(len))
colnames(sum_len)[2]<- paste0(names(Streets_sf_list)[[i]], "_len")
trj_roadways[[i]]<- st_drop_geometry(sum_len)

}


#includes only observations near roadways
trj_roadways_df<- trj_roadways %>% reduce(full_join, by= "ID")

#merge with full trajectory
trj_sf<- trj_sf %>% left_join(trj_roadways_df, by="ID")

#replace NA with 0 (e.g., roadway not near)
trj_sf<- trj_sf%>%
  mutate(across(residential_len:unclassified_len, ~ ifelse(is.na(.), 0, .)))

#create binary indicator (yes/no) roadway is within 100m
trj_sf<- trj_sf%>%
  mutate(across(residential_len:unclassified_len, ~ ifelse(.>0, 1, 0),
                                  .names = "{col}BI"))


#create median split among observations near road
trj_sf<- trj_sf%>%
  mutate(across(c(residential_len:unclassified_len),
          ~ cut2(., cuts = c(0, min(.[.>0]),
            median(.[.>0]),
            max(.[.>0]))), .names = "{col}MED"))
```

Map whether near roadway

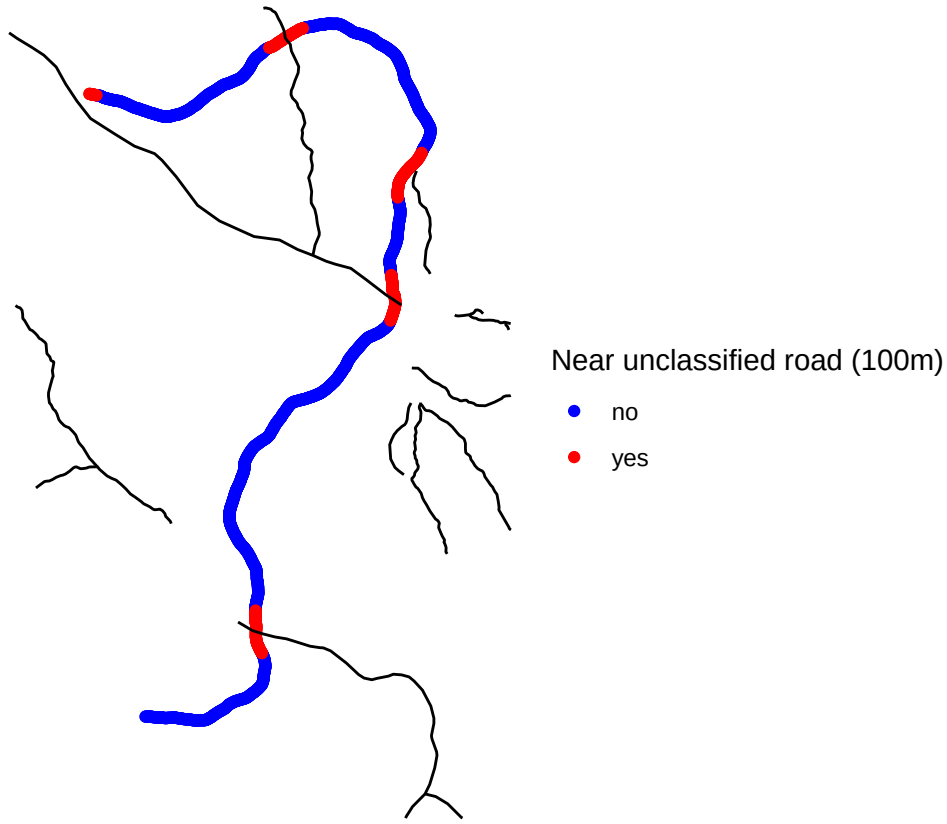
```
ggplot()+
  geom_sf(data=trj_sf, aes(color=as.character(trunk_lenBI)))+
  geom_sf(data = Roadfiltr[Roadfiltr$highway=="trunk",])+
  theme_void()+
  scale_color_manual(name="Near trunk road (100m)",
                     labels=c("no", "yes"),
                     values = c("blue", "red"))
```




```
ggplot()+
  geom_sf(data=trj_sf, aes(color=as.character(residential_lenBI)))+
  geom_sf(data = Roadfiltr[Roadfiltr$highway=="residential",])+
  theme_void()+
  scale_color_manual(name="Near residential road (100m)",
                     labels=c("no", "yes"),
                     values = c("blue", "red"))
```



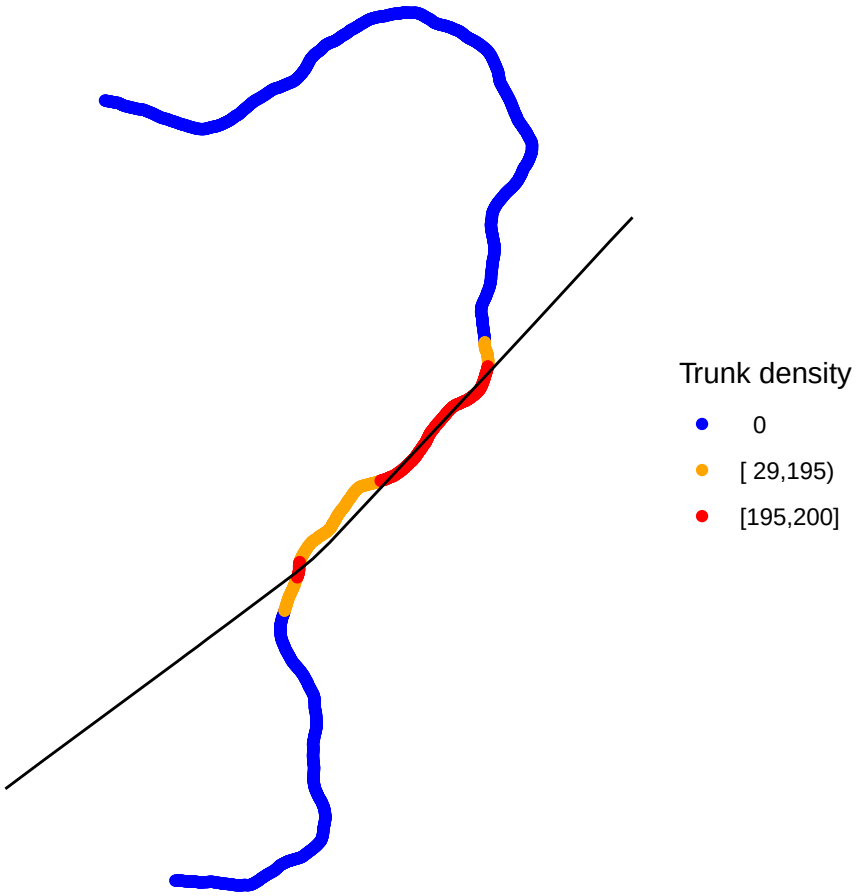

```
ggplot()+
  geom_sf(data=trj_sf, aes(color=as.character(unclassified_lenBI)))+
  geom_sf(data = Roadfiltr[Roadfiltr$highway=="unclassified",])+
  theme_void()+
  scale_color_manual(name="Near unclassified road (100m)",
                     labels=c("no", "yes"),
                     values = c("blue", "red"))
```



Map by roadway density

< or >= median among observations near road

```
ggplot()+
  geom_sf(data=trj_sf, aes(color=as.character(trunk_lenMED)))+
  geom_sf(data = Roadfiltr[Roadfiltr$highway=="trunk",])+
  theme_void()+
  scale_color_manual(name="Trunk density",
                     values = c("blue","orange", "red"))
```


Trunk density
0
[29,195)
[195,200]