Surrounding building density

Danielle Medgyesi

12/20/2022

Open Buildings

Full dataset from Open Buildings for the grid cell containing Ghana (0fd)

https://sites.research.google/open-buildings/

Downloaded on 10/28/2021

Data provided filtered to study region

```
BuildingsBound_sf<- st_read("~/GeospatialGhana/Data/BuildingsBound_sf/BuildingsBound_sf.shp",
                            quiet=TRUE)


#as described in manuscript
#we restricted to buildings with a confidence score of 0.7 or greater
#buildings with lower confidence score prone to inaccuracies

BuildingsBound_sf<- BuildingsBound_sf %>% filter(confdnc>=0.7)


#definitions from Open Buildings
#latitud: latitude of the building polygon centroid,
#longitd: longitude of the building polygon centroid,
#ar_n_mt: area in square meters of the polygon,
#confdnc: confidence score [0.5;1.0] assigned by the model,
#geometry: the building polygon in the WKT format (POLYGON or MULTIPOLYGON),
#fll_pl_: the full Plus Code at the building polygon centroid,


colnames(BuildingsBound_sf)
```

```
## [1] "latitud"  "longitd"  "ar_n_mt"  "confdnc"  "fll_pl_"  "geometry"
```

Simulated GPS trajectory

```
# Create a random trajectory

set.seed(333)
trj <- TrajGenerate(n = 2880, stepLength = 2, angularErrorSd = 0.08, random = TRUE)
trj <- TrajRotate(trj, pi / 1.5, relative = FALSE)
trj<- TrajReverse(trj)
trj<- TrajTranslate(trj, 1700, 2500)
```

```r
#for illustration, use a community centroid as the home coordinates
Home_lon<- -2.453
Home_lat<- 6.85

trj<- trj %>% mutate(x=Home_lon + (x / 6378000) * (180 / pi) / cos(Home_lon * pi/180),
                     y= Home_lat  + (y / 6378000) * (180 / 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]]$y_range+c(-0.001,0.001)
x_range<- ggplot_build(p)$layout$panel_params[[1]]$x_range+c(-0.001,0.001)


#filter building dataset to include only buildings in the region

BB_filter<- BuildingsBound_sf %>%
  filter(between(latitud, y_range[1], y_range[2])) %>%
  filter(between(longitd, x_range[1], x_range[2]))

ggplot()+
  geom_sf(data=trj_sf, size=0.5)+
  geom_sf(data = BB_filter)+
  theme_void()
```

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

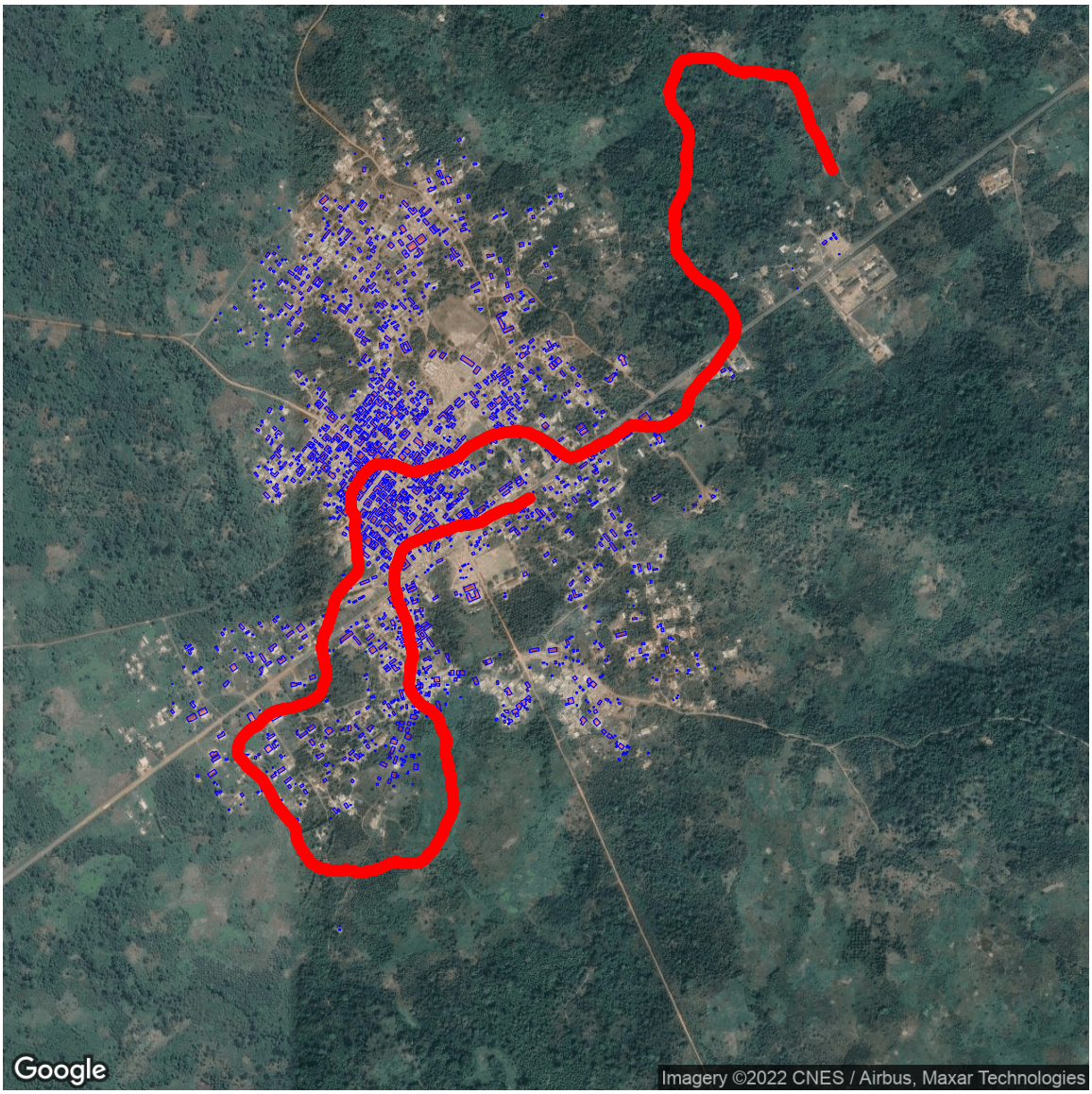
To do so, you can create an account with Google: https://mapsplatform.google.com/

- 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(-2.445, 6.863), zoom=15, maptype = "satellite")


ggmap(base)+
  coord_sf(crs = st_crs(4326)) +
  geom_sf(data = BB_filter, color="blue",
          fill=alpha("#FF0000", .3),
          inherit.aes = FALSE)+
   geom_sf(data = trj_sf,
          color="red",
          inherit.aes = FALSE)+
  theme(axis.line = element_blank(),
        axis.text = element_blank(),
```

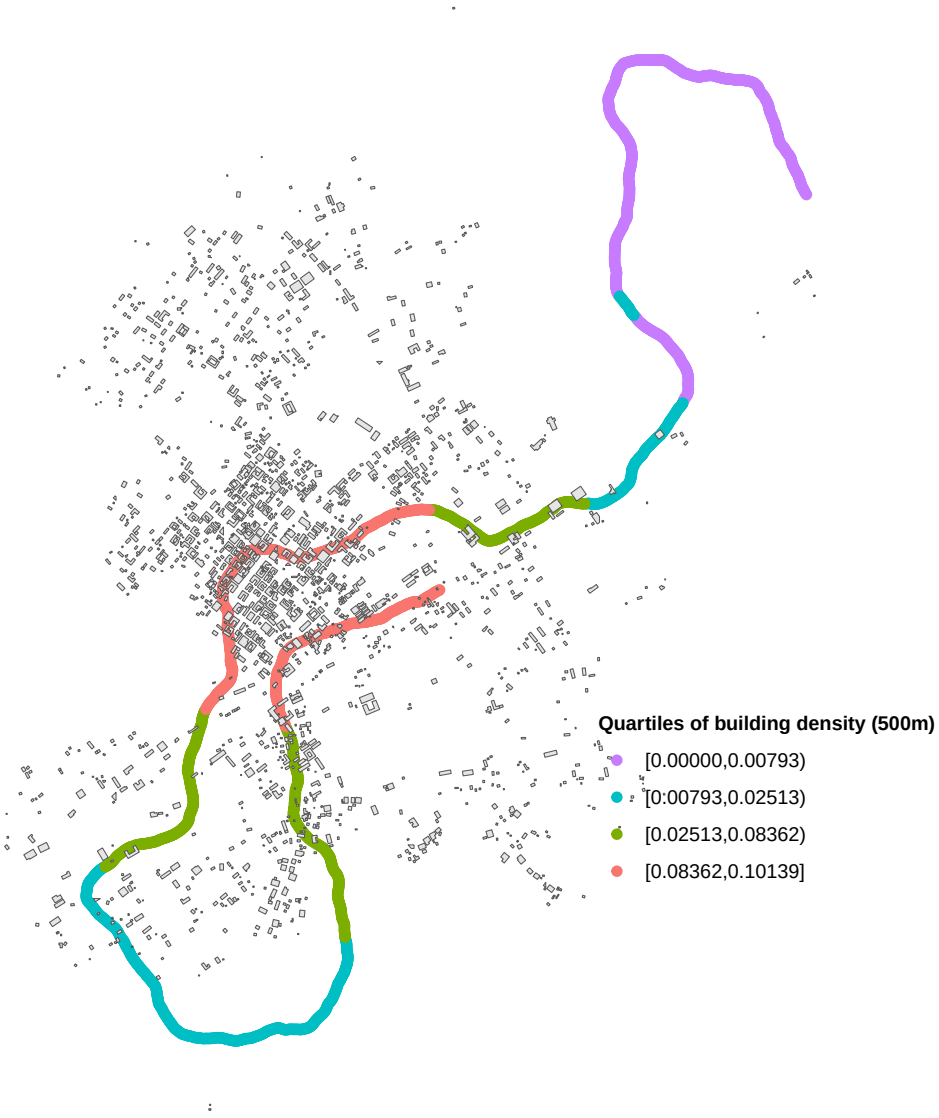
```
        axis.ticks = element_blank(),
        plot.margin = unit(c(0, 0, -1, -1), 'lines')) +
  xlab('') +
  ylab('')
```



Estimate surrounding building density

```
#Transform to Ghana crs
#https://epsg.io/?q=Ghana

trj_sf<- trj_sf %>% st_transform(32630)
BB_filter<- BB_filter %>% st_transform(32630)

# Create 500m buffer around each trajectory point
trj_sf500 <- st_buffer(trj_sf, dist = 500)
```

```r
#identify buildings that intersect each 500m buffer
trj_build_int<- st_intersects(trj_sf500, BB_filter)


traj_build_list<- list()
for (i in 1:length(trj_build_int)) {
  #count number of buildings within 500m
  N_build<- length(trj_build_int[[i]])
  #sum area of buildings within 500m
  Area_build<- sum(st_area(BB_filter[1:nrow(BB_filter)%in% trj_build_int[[i]],]))
  #calculate density
  Density_Build<- as.numeric(Area_build)/((500^2)*pi)
  traj_build_list[[i]]<- bind_cols(N_build=N_build,
                                   Area_build=Area_build,
                                   Density_Build=Density_Build)

}

traj_build_df<- rbindlist(traj_build_list)


trj_sf<- cbind(trj_sf, traj_build_df)

traj_sf<- trj_sf %>% mutate(DensityQ=as.character(cut2(Density_Build,g=4)))

DensQ<- traj_sf %>% st_drop_geometry() %>% dplyr::select(DensityQ)

ggplot()+
  geom_sf(data=trj_sf, aes(color=DensQ$DensityQ))+
  geom_sf(data = BB_filter)+
  theme_void()+
  theme(legend.position = c(0.9, 0.3), legend.text=element_text(size=8),
        legend.title =element_text(size=8, face = "bold"),
       legend.key.size = unit(15, "points"))+
  scale_color_manual(name="Quartiles of building density (500m)",
                     values = c("#C77CFF", "#00BFC4", "#7CAE00", "#F8766D"))+
    labs(caption = "Among buildings with confidence score of 70% or greater")
```

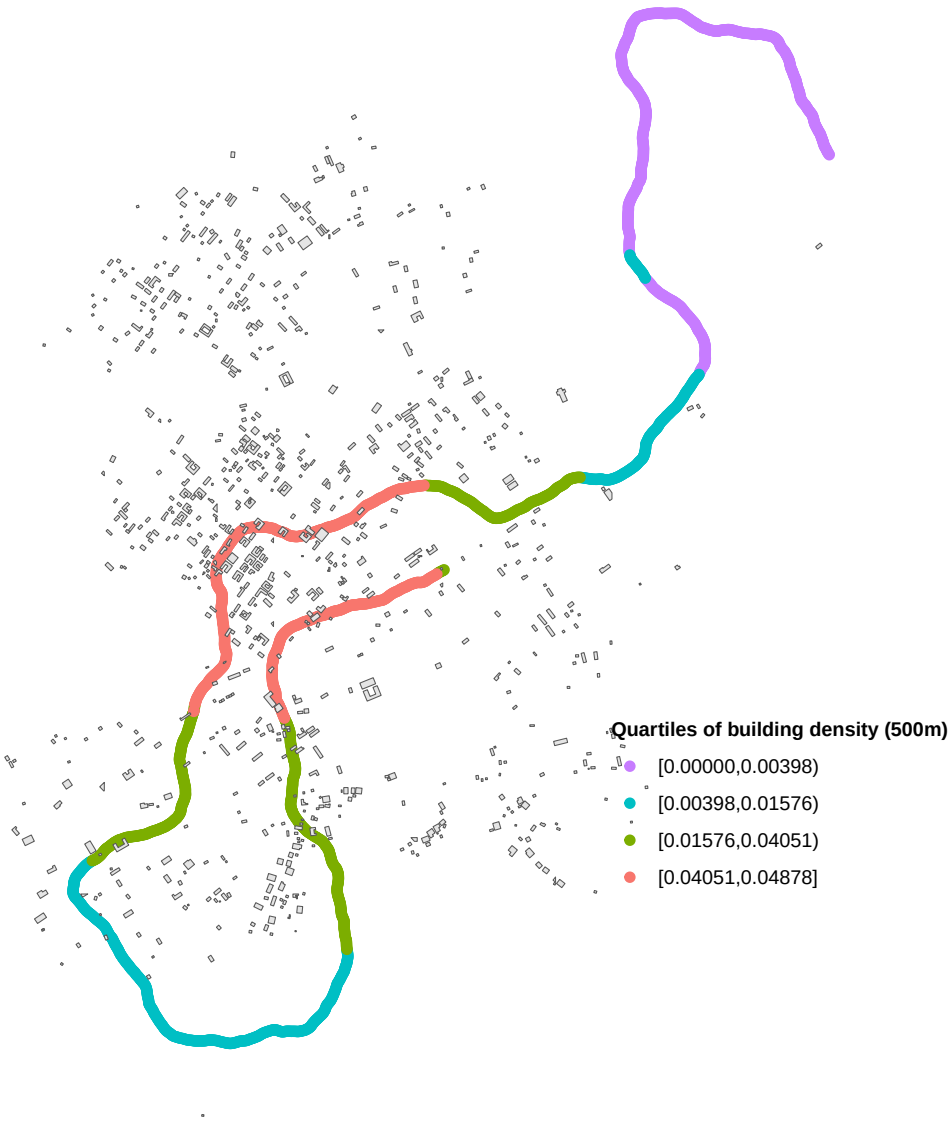


Among buildings with confidence score of 70% or greater

Restrict to buildings with 80%+ confidence score, which tend to be larger buildings

```
BB_filter80<- BB_filter %>% filter(confdnc>=0.8)

#identify buildings that intersect each 500m buffer
trj_build_int80<- st_intersects(trj_sf500, BB_filter80)


traj_build_list80<- list()
for (i in 1:length(trj_build_int80)) {
  #count number of buildings within 500m
  N_build80<- length(trj_build_int80[[i]])
  #sum area of buildings within 500m
  Area_build80<- sum(st_area(BB_filter80[1:nrow(BB_filter80)%in% trj_build_int80[[i]],]))
```

```
  #calculate density
  Density_Build80<- as.numeric(Area_build80)/((500^2)*pi)
  traj_build_list80[[i]]<- bind_cols(N_build80=N_build80,
                                     Area_build80=Area_build80,
                                     Density_Build80=Density_Build80)

}

traj_build_df80<- rbindlist(traj_build_list80)


trj_sf<- cbind(trj_sf, traj_build_df80)

traj_sf<- trj_sf %>% mutate(Density80Q=as.character(cut2(Density_Build80,g=4)))

Dens80Q<- traj_sf %>% st_drop_geometry() %>% dplyr::select(Density80Q)

ggplot()+
  geom_sf(data=trj_sf, aes(color=Dens80Q$Density80Q))+
  geom_sf(data = BB_filter80)+
  theme_void()+
  theme(legend.position = c(0.9, 0.3), legend.text=element_text(size=8),
        legend.title =element_text(size=8, face = "bold"),
       legend.key.size = unit(15, "points"))+
  scale_color_manual(name="Quartiles of building density (500m)",
                     values = c("#C77CFF", "#00BFC4", "#7CAE00", "#F8766D")) +
  labs(caption = "Among buildings with confidence score of 80% or greater")
```



Among buildings with confidence score of 80% or greater