Delimitation of hospital catchment areas

Illustration with the hospitals of Belgium

Jérémie Bihin, Simon Dellicour, Catherine Linard

May 10, 2021

The COVID-19 pandemic is affecting nations globally, but with an impact exhibiting significant spatial and temporal variation at the sub-national level. Identifying and disentangling the drivers of resulting hospitalisation incidence at the local scale is key to predict, mitigate and manage epidemic surges, but also to develop targeted measures. However, this type of analysis is often not possible because of the lack of spatially explicit health data and spatial uncertainties associated with infection. To overcome these limitations, we propose to work at the level of hospital catchment areas (HCAs). The hypothesis underlying the concept of HCA is that patients normally choose the hospital closest to their home. While the simplest approximation of spatial accessibility is Euclidean distance, a more realistic way to generate accessibility maps is to estimate travel times between populations and hospitals. In practice, we propose to use use the friction surface developed by Weiss *et al.* [1] to derive maps of travel time between any 1 km^2 pixel in Belgium and each of the 103 hospitals. Each pixel is here attributed to the hospital that minimised such travel time, and the set of pixels attributed to a given hospital was defined as its catchment area. In areas where multiple hospitals could be found within the same pixel of the friction surface, catchment areas and associated epidemiological data were aggregated ($n = 2$).

The preliminary step is to (install and) load the following R packages:

```
> library(gdistance)
> library(malariaAtlas)
> library(raster)
> library(rgdal)
> library(sf)
```

Step 1: loading the different GIS input files

For this tutorial, we will need to load two distinct input files: the friction raster ("Friction_raster_Belgium.tif") and a shapefile gathering the geographic position of the 103 hospitals considered here ("All_hospitals_Belgium.shp", Fig. 1A):

```
> frictionR = raster("Friction_raster_Belgium.tif")
> hospitals = read_sf("All_hospitals_Belgium/All_hospitals_Belgium.shp")
```

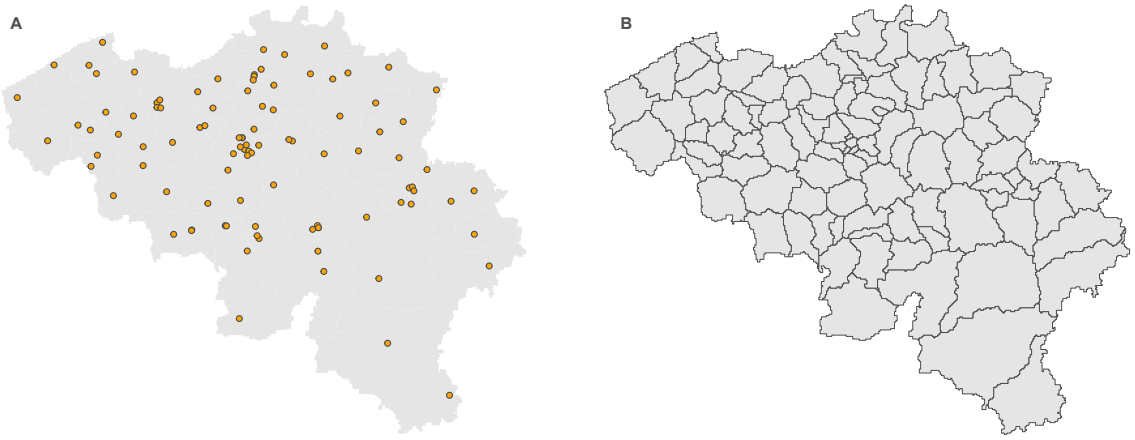

Figure 1: geographic position of the 103 Belgian hospitals considered here (**A**) and the resulting delimitation of hospital catchment areas (**B**, final shapefile output).

Step 2: computing the travel time to the closest hospital for the whole country

The second step consists in generating a raster map in which each pixels is associated with an estimation of the travel time to the closest hospital for the whole country. For this purpose, we use the least-cost path model implemented in the function "accCost" of the R package "gdistance":

```
> tr = transition(frictionR, function(x) 1/mean(x), 8)
> trG = geoCorrection(tr)
> hospitals_xy = st_coordinates(hospitals)
> hospitals_access = accCost(trG, hospitals_xy)
```

The resulting raster map is displayed in Figure 2A, which becomes the raster of reference for the following steps.

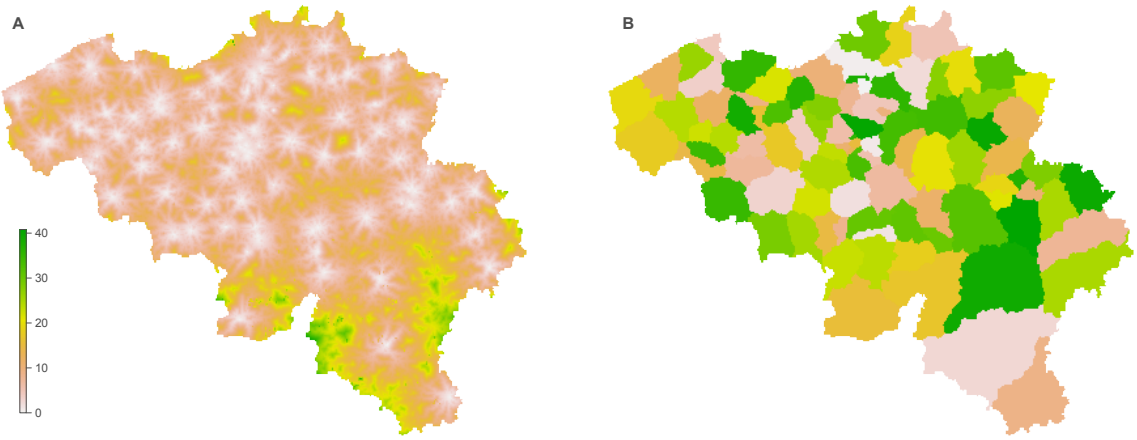

Figure 2: travel time (in minutes) to the nearest hospital (**A**) and the resulting delimitation of hospital catchment areas (**B**, intermediate raster output).

Step 3: for each hospital, estimating the travel time from each pixel of the map

For each hospital, we will now estimate the travel time from each pixel of the map, i.e. computing the least-cost distance map for each hospital. For this third step, we will again use the least-cost path algorithm implemented in the R package "gdistance":

```
> s = stack()
> for (i in 1:nrow(hospitals)) {
>  coord = st_coordinates(hospitals[i,])
>  rast = accCost(trG, coord); s = stack(s, rast)
>  print(paste0("Computation for hospital ",i))
> }
> vect = getValues(hospitals_access)[!is.infinite(getValues(hospitals_access))]
> mat = matrix(nrow=nrow(hospitals), ncol=length(vect))
> for (i in 1:nrow(hospitals)) {
>  mat[i,] = getValues(s[[i]])[!is.infinite(getValues(s[[i]]))]
> }
```

After this step, we end up with a matrix "mat" that summarises, for each hospital (row), the travel time from any pixel of the Belgian map (column).

Step 4: for each pixel, identifying which hospital is associated with the lowest travel time

The fourth step consists in identifying, for each pixel, which hospital is associated with the lowest travel time. For this purpose, we simply analyse the matrix "mat" generated in step 3 to extract the lowest travel time and related hopsital:

```
> vectsel = c()
> for (i in 1:ncol(mat)) {
>  vectsel[i] = which(mat[,i]==min(mat[,i]))
> }
> vectInf = values(hospitals_access); j = 1
> for (i in 1:(ncol(hospitals_access)*nrow(hospitals_access))) {
>  if (is.infinite(vectInf[i]) == FALSE) {
>  vectInf[i] = vectsel[j]
>  j = j+1
>  }
> }
```

We then save the extracted data within an intermediate raster file (displayed in Figure 2B):

```
> catchmentAreas = raster(vals=(vectInf), ext=extent(hospitals_access),
  crs=crs(hospitals_access), nrows=dim(hospitals_access)[1],
  ncols=dim(hospitals_access)[2])
```

Step 5: converting and saving the resulting raster into a shapefile gathering HCAs

The final step consists in transforming the intermediate raster file into a shapefile object in which each polygon is associated with two different metadata: the area of the polygon and the ID(s) of hospitals it contains. For this purpose, we run the following script:

```
> catchmentAreas = rasterToPolygons(catchmentAreas, dissolve=T, na.rm=T)
> catchmentAreas = subset(catchmentAreas, is.finite(catchmentAreas@data[,"layer"]))
> metadata = matrix(nrow=dim(catchmentAreas@data)[1], ncol=2)
> colnames(metadata) = c("area","X_ID")
> xS = as(hospitals,"Spatial")@coords[,1]; yS = as(hospitals,"Spatial")@coords[,2]
> for (i in 1:dim(metadata)[1]) {
>  area = 0
>  for (j in 1:length(catchmentAreas@polygons[[i]]@Polygons)) {
>  pol = catchmentAreas@polygons[[i]]@Polygons[[j]]
>  indices = which(point.in.polygon(xS, yS, pol@coords[,1], pol@coords[,2]) == 1)
>  if (length(indices) > 0) {
>  metadata[i,"X_ID"] = hospitals$ID[indices[1]]
>  if (length(indices) > 1) {
>  for (k in 2:length(indices)) {
>  metadata[i,"X_ID"] = paste(metadata[i,"X_ID"],hospitals$ID[indices[k]],sep="-")
>  }}}
>  p = Polygon(pol@coords); ps = Polygons(list(p),1); sps = SpatialPolygons(list(ps))
>  pol = sps; proj4string(pol) = crs(catchmentAreas)
>  area = area + raster::area(pol)
>  }
>  metadata[i,"area"] = round(area)
> }
> catchmentAreas@data = as.data.frame(metadata)
> writeOGR(catchmentAreas, dsn="Hosp_catchmentArea", "Hospital_catchment_areas",
  driver="ESRI Shapefile")
```

The resulting shapefile is displayed in Figure 1B.

References

[1] Weiss DJ, Nelson A, Gibson HS, Temperley W, Peedell S, Lieber A, Hancher M, Poyart E, Belchior S, Fullman N, Mappin B, Dalrymple U, Rozier J, Lucas TCD, Howes RE, Tusting LS, Kang SY, Cameron E, Bisanzio D, Battle KE, Bhatt S, Gething PW (2018). A global map of travel time to cities to assess inequalities in accessibility in 2015. *Nature* 553: 333.