Exercise S

Geocomputation of various indices

In this exercise, we will use several algorithms from different open GIS software, including QGIS, SAGA (for raster analysis) and Grass (for network analysis). All needed algorithms are loaded in QGIS, but you need to open the version "with Grass". All needed data are available on Moodle:

https://enacshare.epfl.ch/fyp9sQi7wYJhVUvu6cGNT5rWzjKeSHa

The layers you need for this week are

- subset individual data points: bmi_subset.shp
- network: GMO_GRAPHE_ROUTIER_21781.shp
- green spaces: greenspace.shp
- bus stops: TPG_ARRET_21781.shp
- DEM: swissalti3D2016.tif

0) Prequisite

This exercise contains complex geographic computation and therefore it is required that all your layer share the same projection. Therefore, choose a common projection - all your layer should already be in EPSG 21781 but if this was not the case reproject the layers that do not match the right projection (right click, save as).

The individual layer you have (bmi_ssb_15423.xlsx) contains a lot of points, which will slow down the analysis. For this exercise, we propose that you use a random subset of the input individual layer (bmi_subset.shp). Compared to the initial layer, it also has an additional id column. Afterwards, if you wish to use some of the analysis presented here for your project, you can redo part of the analysis with the initial entire layer.

1) Buffer

Buffers are useful to describe the environment in which an individual leaves.

Calculate the percentage of the area covered by a green space in a buffer of 1000m around each individual from the subset layer.

- Create a buffer around each individual (Vector>Geoprocessing Tool>Fixed distance Buffer).
- Create the intersection of the buffer with the green spaces (Vector > Geoprocessing Tools > Intersection)
- Using the field calculator, add a column specifying the area of each intersected polygon.
- Save your layer as a shapefile if your layer is a temporary one
- Calculate the sum of the area for each individual id: Use Layer > Add Layer > Add/Edit virtual layer. Using the import button, import your intersection layer, type the query "SELECT id, sum(area) from intersection group by id" (change the name of the area attribute and intersection layer)

- Add a column to the initial subset individual layer corresponding to the summed area (right click, properties, tab join and create the corresponding join with your virtual layer)
- Calculate the percentage by dividing the summed area by the area of the area of the buffer (calculate it by hand knowing that it is a circle with radius 1000m)

2) Nearest feature

If you have to calculate the minimum distance between two sets of points, you can use the Vector > Analysis Tools > Distance matrix. If you are working with lines or polygons, you will need to use a plugin.

Using the NNJoin plugin (once installed, available in the vector menu), calculate for all individuals the nearest distance to a green area.

3) DEM analysis

Digital Elevation models are very informative layers, since many indicators can be computed from them. SAGA (integrated in QGIS) is a powerful tool to compute many of them. Here we will focus only on the slope.

- Slope along a line

You can use SAGA to create a slope raster. However here, we are interested in the profile along the road. Use the SAGA algorithm in the processing toolbox (if the toolbox does not show up on the right, go to treatment>Toolbox) named Terrain analysis – Profile > Profile from lines. Make sure to uncheck "Each line as new profile". This will segment your streets into equally spaced points with corresponding altitude on the used DEM.

Next we need to calculate the average slope of each line. Careful, this does not correspond to the difference between the beginning and ending of the line divided by the distance. This would not account for uphill followed by downhill paths or inversely. Instead we need to calculate the cumulative change in altitude in absolute value. Export your profile as csv and use the following matlab script (you are free to use any other software if you feel more comfortable with). First open your csv in notepad and look at its structure. Each line is identified by an incremental LINE_ID and each point has a unique ID. The X, Y and Z coordinate of each point is given.

Once in matlab, import your profile csv file using the "import Data" tool. Keep the default "Column vectors" for import format. Then run the following script

```
ligne=0;
dz_cum=0;
avg_slope_array=zeros(max(LINE_ID)+1, 3); %first column, avg slope gradient
      %second and third column: X and Y coordinate of the penultimate point of
      %the line (the last point might be at an intersection with another line
      %Note that the LINE_ID begins with zero, which explains the +1
for point=2:max(ID)
    if LINE_ID(point)~=ligne
        avg_slope_array(ligne+1,1)=dz_cum/DIST(point-1); %slope: dz/dx
        avg_slope_array(ligne+1,2)=X(point-2); %we save the penultimate
      %point, because the very last point might be at the intersection with
      %another line, and the join afterwards will be ambiguous
```

```
        avg_slope_array(ligne+1,3)=Y(point-2);
        ligne=LINE_ID(point);
        dz_cum=0;
    else
        dz_cum=dz_cum+abs(Z(point)-Z(point-1)); %cumulative change in altitude
                                                %in absolute value
    end
end
avg_slope_array(max(LINE_ID)+1)=dz_cum/DIST(max(ID)); %update last line
avg_slope_array(max(LINE_ID),2)=X(max(ID)-2);
avg_slope_array(max(LINE_ID),3)=Y(max(ID)-2);

%write header to file
fid = fopen('avg_slope.csv','w');
textHeader='avg_slope,X,Y';
fprintf(fid,'%s\n',textHeader);
fclose(fid);
%write data to end of file with enough precision
dlmwrite('avg_slope.csv',avg_slope_array,'-append','precision',12);
```

Once run, you can quit matlab and import your csv into QGIS (using Layer > Add Layer > Add delimited text layer so as to specify the X and Y column) and set the layer CRS (EPSG 21781). We now need to link those points with the lines of the network. Careful, the points might be slightly shifted (due to loss of digits in the process) and might not intersect perfectly the line so that the "join attribute by location" would not work in this case. Instead use the NNJoin plugin. In the end you should have a layer with the initial network and an additional column (among other) specifying the average slope of the road.

- **Walking time along a line**

To this end, we can use Tobbler's equation that calculates the walking speed according to the slope

$$Walking\ speed\ \left(\frac{km}{h}\right) = 6 \cdot e^{-3.5 \cdot abs(S+0.05)}$$

Where S is the slope in gradient. We would like the average speed for up and downhill, thus, the formula becomes (at the same time transforming the unit into m/min)

$$Walking\ speed\ \left(\frac{m}{min}\right) = \frac{6 \cdot e^{-3.5 \cdot abs(S+0.05)} + 6 \cdot e^{-3.5 \cdot abs(-S+0.05)}}{2} \cdot \frac{1000}{60}$$

The walking time (length/walking speed), will become

$$Walking\ time\ (min) = \frac{line\ length}{\frac{6 \cdot e^{-3.5 \cdot abs(S+0.05)} + 6 \cdot e^{-3.5 \cdot abs(-S+0.05)}}{2} \cdot \frac{1000}{60}}$$

Create a new field using the field calculator corresponding to the time needed to cross the road segment (the length in meter is already calculated in a field called SHAPE_LEN).

Note: You can improve the cost you calculated by taking into account the type of road and its pedestrian friendliness. For the moment, the cost you calculated is the time it takes to cross the road

segment. You could increase your cost of cantonal and national roads, so that the shortest path algorithm will avoid taking these roads.

4) Network analysis

Network analysis can easily be computed using Grass (available within QGIS in the processing toolbox). Note that if your network is not prepared, you should use v.clean to prepare your network. Most importantly, v.clean will split lines at intersections. In our network, it is already the case except for the roads that cross on a 2D surface but that do not intersect in reality (e.g. bridge on a road) and we don't want those lines to be split. Here we will only explore the distances along a network but keep in mind that you can use GRASS to calculate centrality, connectivity and many more measures associate to road networks.

- **Distance/time along a network**

Using v.net.distance, compute the distance along the network from the individuals to the nearest bus stop of all individuals of the subset layer. Increase the threshold (for connecting centers to the network) to 100m. Then, in the advanced parameters, set the "Arc forward/both direction(s) cost column" as the walking time you calculated. Leave all other parameters as default. This outputs the shortest path from your individuals to the nearest bus stop. The field "dist" of this line layer contains the time required to reach the nearest bus stop (if you had not set the cost it would contain the distance). Use the "NNJoin" plugin to assign this distance to each point in the individual layer.

- **Isochrones**

Isochrone maps are map showing the time needed to reach a facility of interest. Here we will focus on the accessibility to bus stop.

To this end, use the grass function v.net.iso. Specify a point layer (bus stop) and a network layer (gmo…). Again, in the advanced parameters, set the "Arc forward/both direction(s) cost column" as before. Calculate isochrones for 1,2,5,10 minutes (if you have correctly computed your walking time, you can directly set the above mentioned number in the "Costs for isoline" field).

This outputs the classification of all segments of the network in categories.