Appendix 1 – Download OpenStreetMap (OSM) data

Why OSM?

* For the analysis of the walkable environment of European cities, we needed available and comparable data on the network that connects the residential buildings with the public green spaces.
* To ensure a high resolution of the analysis, we needed data on the residential buildings, as well.
* OSM offers global coverage with varying data density.
* Fortunately, the reliability of OSM data is usually higher in larger cities.
* Since we wanted to develop a workflow using free and open source data that produces comparable results for all of Europe, OSM was the only available choice.

The OSM project:

* OSM is a community-based project that provides free geospatial data.
* The OSM community seeks to create a database of the entire planet that is free and editable.
* For creation and verification of the OSM map, the community uses a wide variety of data sources.
* Among these sources are aerial photographs, GPS-devices and maps.
* The OSM community consists of a variety of contributors, ranging from enthusiastic mappers to GIS-professionals and engineers.
* Registration is mandatory for editing the OSM map. (<https://www.openstreetmap.org/about>).
* At the time of writing this paper, the number of registered OSM users is about 8.5 million. (<https://planet.openstreetmap.org/statistics/data_stats.html>)
* In 2022, the entire OSM dataset contains a total of roughly 7.7 billion nodes and about 860 million ways.
* OSM follows an *“open data”* policy, meaning that the data can be used for any purpose, as long as OSM and its contributors are mentioned (https://www.openstreetmap.org/about).

Downloading OSM data:

* Downloading larger chunks of OSM data can prove difficult.
* For downloading OSM data, the OSM community offers the Overpass API on different public instances (https://wiki.openstreetmap.org/wiki/Overpass\_API).
* Since OSM is an open source project, all servers that provide OSM data are considered public goods.
* Heavy usage of the Overpass API has to be avoided and should not surpass 10.000 requests per day or 1 GB download volume (https://dev.overpass-api.de/overpass-doc/en/preface/commons.html).
* If over-use of the servers is detected, a user will usually be timed out (https://dev.overpass-api.de/overpass-doc/en/preface/commons.html).
* To download OSM data in accordance to the community guidelines, we created a (yet unpublished) R package that automatizes downloading OpenStreetMap network and building data on a city level.

Functionality:

* The downloading-workflow consists of the download\_OSM function and various subfunctions and relies heavily on the osmdata package.
* Required input for the download\_OSM function is the city code (URAU-code) of the desired city and the directory containing the file with the city boundaries.
* The workflow follows the steps: subdividing the city, downloading the data, cleaning the data and writing the data to file.

1. Subdividing the city
   * We have taken several precautions to limit the number of requests and the overall download size of each request.
   * As a first measure, the download\_OSM-function extracts the city boundary that corresponds to the provided URAU-code.
   * The city boundary will be cut into a grid of boundary boxes with 2 km edge length.
   * Larger sized boundary boxes have shown to produce too large data chunks.
   * Especially in dense parts of large cities these large data chunks led to queries that were frequently canceled by the Overpass API.
   * Smaller boundary boxes, on the other hand, created an unnecessarily high number of queries.
   * A smaller edge length of the boundary boxes would also cause more duplicates on the edges of the boxes.
2. Downloading process
   * During the downloading process, R will try to download the OSM data for each of the boundary boxes individually.
   * For each of the boundary boxes, R will communicate with the different instances of the Overpass API.
   * If each instance offers the same number of slots, one will be chosen randomly.
   * Otherwise, R will choose (one of) the Overpass API instances with the highest number of available slots.
   * If no slots are available, R will timeout for 30 seconds and restart communication with the Overpass API afterwards.
   * The chosen Overpass API instance will be set via the set\_overpass\_url- function from the osmdata package.
   * In the next step, R will create an overpass-query using the osmdata opq-function.
   * R will create individual queries for the building and the network data, i.e. create two separate download requests.
   * With the created queries, R will try to download the OSM data that is located inside the boundary box from the set Overpass API instance.
3. Cleaning the data
   * Once the OSM data is downloaded to the computers RAM, R will try to ensure the integrity of the data.
   * First, only the columns matching the string “building$” (for buildings) or “highway” (for network data) will be selected.
   * Previous attempts have shown that several non UTF-8 characters in the column names of the OSM data are not compatible with the Geopackage (.gpkg) format.
   * In the second cleaning step, R will cast the geometries to polygons (for buildings) or linestrings (for network data).
   * The OSM data is provided in individual layers for each geometry class (point, linestring, polygon, multilinestring, multipolygon).
   * This step will first and foremost ensure the data’s compatibility across different R packages and functions.
   * In addition to the compatibility, casting the geometries to a lower level will also prevent erroneous geometries from causing trouble down the workflow.
   * OSM is a large and diverse dataset with only the community validating the correctness of the data - so errors have to be expected.
4. Writing the data to file
   * Finally, R will generate an output directory based on the input directory and the city code.
   * If the OSM data consisted of multiple layers with different geometry classes, the now homogenized data will be appended to the same file and the same layer.

* These steps will be repeated until the OSM data in each boundary box is downloaded.
* The individual subfunctions that are being called by the download\_OSM function can be called separately by a skilled user, as well.