Motorway Data Set Construction from Scratch

Brown text indicates folder name, green text indicates file names. Files without ending denote shape (.shp) files with their corresponding files (.dbf, .prj, .shx, …).

1. **Setup**
   Clone https://github.com/maikemp/motorwayData
   Add a folder called data to the root directory, it is ignored by the repository. data should contain three sub-directories: data_in, data_temp, and data_out.
2. **Data Download**
   Download all input data and store it in the following structure in sub-directories of data_in:

Source	Name	Link	Save format
OSM	**OpenStreetMap** motorway network	Download interface	data_in → overpass_turbo with files: ➢ motorways_DE_2017-07-01.geojson ➢ motorways_DE_2018-07-01.geojson ➢ motorways_DE_2019-07-01.geojson

	Instructions:
	1. Enter the following query for each year:
	`[date:"year-07-01T00:00:00Z"][timeout:6000];`
	`area[boundary=administrative]["ISO3166-2"~"^DE-"];`
	`way(area)[highway=motorway];`
	`(._;>;);`
	`out;`
	2. In the Window "Große Datenmengen", click "Trotzdem weitermachen" and wait for shapes to appear in the map.
	3. Download via "Export" – "GeoJSON: download"
	Note: queries can take a while to process (~30 minutes). There may also be limitations on the amount of data one can request on a single day. Therefore, it is best to **calculate with several days to finish all data downloads from the overpass API**.		

OSM	**OpenStreetMap** all roads to extract side roads from it.	Collection of downloads for all federal states.	data_in → OSM_shape_BL with subfolders: ➢ baden-wuerttemberg-latest-free ➢ bayern-latest-free ➢ brandenburg-latest-free ➢ bremen-latest-free ➢ hamburg-latest-free ➢ hessen-latest-free ➢ mecklenburg-vorpommern-latest-free ➢ niedersachsen-latest-free ➢ nordrhein-westfalen-latest-free ➢ rheinland-pfalz-latest-free ➢ saarland-latest-free ➢ sachsen-anhalt-latest-free ➢ sachsen-latest-free ➢ schleswig-holstein-latest-free ➢ thueringen-latest-free
	Instructions: from Sub Region table, download via Quick Link under .shp.zip for all states, except for Berlin (included in Brandenburg). Unpack ZIP files into folder structure on the right. Only files that need to be kept within these subfolders are all the files starting with → gis_osm_roads_free_1 (.shp files and corresponding files.) Note: these links do not seem to be available anymore for the states Baden-Württemberg, Bayern, and Nordrhein-Westfalen. Instead, click on these states, and download the .shp.zip files for all sub-regions.		
OSM	**OpenStreetMap** motorway links	Download interface	data_in → overpass_turbo with files: ➢ motorways_links_2019-07-01.geojson

	Instructions:
	1. Enter the following query:
	`[date:"2019-07-01T00:00:00Z"][timeout:6000];`
	`area[boundary=administrative]["ISO3166-2"~"^DE-"];`
	`way(area)[highway=motorway_link];`
	`(._;>;);`
	`out;`
	2. In the Window "Große Datenmengen", click "Trotzdem weitermachen" and wait for shapes to appear in the map.
	3. Download via "Export" – "GeoJSON" – "download"		

Statistische Ämter des Bundes und der Länder	**Unfallatlas** - locations of injury crashes	Link	data_in → unfallatlas with subfolders: ➢ Unfallorte_2017_Shapefile → Unfallorte2017_LinRef ➢ Unfallorte_2018_Shapefile → Unfallorte2018_LinRef ➢ Unfallorte_2019_Shapefile → Unfallorte2019_LinRef
	Instructions: click on "Unfallatlas und OpenData" → in pop-up window, click "Download Unfallorte year – shapefile (zip)" for each year. Unpack ZIP files into folder structure on the right (with no additional sub-folders).		
BASt	**BISStra** - Federal highway network	Link	data_in → BISStra ➢ BFStr_Netz_NK.geojson ➢ BFStr_Netz_SK.geojson
	Unrequired file: BFStr_Netz_NP.geojson		

<table>
<tr><td></td><td colspan="2"></td><td>Note: only the most recent version available. Version used in original data set construction: 2019q4.</td></tr>
<tr><td rowspan="2">BASt</td><td>Automatic counting stations 2017-2019</td><td>Link</td><td rowspan="2">data_in → counting_stations
➢ Jawe2017.csv
➢ Jawe2018.csv
➢ Jawe2019.csv
Note: data has been updated, using newer versions may slightly differ from data used for the original data set construction.</td></tr>
<tr><td colspan="2">Instructions:
1. Under “Ergebnisse früherer Jahre”, select the respective years.
2. Under “Straßenklasse”, select “A”.
3. Click “Als CSV-Datei exportieren”.</td></tr>
<tr><td rowspan="2">BASt</td><td>Extrapolation of traffic counts of 2015 and of temporary measurements of 2016 to 2019.</td><td>Link</td><td rowspan="2">data_in → counting_stations
➢ Fortschreibung_BAB_15_to_19.xlsx</td></tr>
<tr><td colspan="2">Instructions: under “Fortschreibung/Hochrechnung der Ergebnisse der SVZ 2015 […] 2019”, select “Ergebnisse der Bundesautobahnen (xlsx, 5MB)”.</td></tr>
<tr><td rowspan="2">NASA</td><td>SRTM - Shuttle Radar Topography Mission 1 arc second</td><td>Download interface</td><td rowspan="2">data_in → NASA_SRTM30_tiles
➢ N47E007.hgt
➢ …
➢ N54E011.hgt</td></tr>
<tr><td colspan="2">Instructions:
1. Create a NASA Earthdata Login
2. Individually select tiles that cover (parts of) Germany and download via „Download DEM“</td></tr>
<tr><td rowspan="2">DWD</td><td>Climate Data Center, annual weather/climate grids</td><td>Link and wind data link</td><td rowspan="2">data_in → DWD with subfolders:
➢ air_temperature_mean
→ TAMM_17_year_01.asc
➢ frost_days
→ TADNCDLT00_17_year_01.asc
➢ ice_days
→ TADXCDLT00_17_year_01.asc
➢ precipGE10mm_days
→ RRDSCDGE10_17_year_01.asc
➢ precipGE20mm_days
→ RRDSCDGE20_17_year_01.asc
➢ precipGE30mm_days
→ raster_rrshs_year_jahr.asc
➢ precipitation
→ RSMS_17_year_01.asc
➢ snowcover_days
→ SHDSCDGE01_17_year_01.asc
➢ summer_days
→ TADXCDGE25_17_year_01.asc
➢ sunshine_duration
→ SDMS_17_year_01.asc
➢ wind
→ wind_wdat_geo_10m_BRD_200m.asc</td></tr>
<tr><td colspan="2">Instructions:
1. Click on each of the weather characteristics (listed on the right) individually (Note: wind not in the list, download later).
2. Select and download grids for the respective years.
3. Download wind_wdat_geo_10m_BRD_200m.asc.zip from the wind data link.
4. Unpack all data into folder structure on the right. The “year” in the file name indicates is a placeholder for the respective years.</td></tr>
<tr><td rowspan="2">BBSR</td><td>INKAR - indicators and maps of spatial and urban development</td><td>Link</td><td rowspan="2">data_in → INKAR
➢ inkar_extract.csv
Note: only the most recent version is downloaded through this procedure. Data used for original data set construction refers to the year 2017.</td></tr>
<tr><td colspan="2">Instructions:
1. Click “INKAR starten” → “Weiter >”
2. Select the following indicators from categories (by clicking on indicator and then “Auswählen”):
• Beschäftigung und Erwerbstätigkeit – Struktur – Erwerbsquote
• Bevölkerung – Altersstruktur – Einwohner von 18 bis unter 25 Jahren; Einwohner 65 Jahre und älter
• Bevölkerung – Bevölkerungsstruktur – Frauenanteil
• Privateinkommen, Private Schulden – Haushaltseinkommen
• Siedlungsstruktur – Ländlichkeit, Einwohnerdichte
• Verkehr und Erreichbarkeit – Straßenverkehr – Pkw-Dichte
• Wirtschaft – Wirtschaftliche Leistung – Bruttoinlandsprodukt je Einwohner
3. Click “Weiter >”, tick “alle” at the top of the list, click “Fertigstellen”, select “Tabelle exportieren”, in pop-up window select “CSV” to download data.</td></tr>
</table>

BKG	**VG250-EW 31.12**.- Administrative areas 1:250 000 with population numbers	Link	data_in → BKG → vg250-ew_ebenen_1231 ➢ VG250_KRS
	Instructions: Download „vg250-ew_12-31.utm32s.shape.ebenen.zip“ and unpack into folder. **Note**: Only need to keep files starting with VG250_KRS (shape file and corresponding files).		
BASt	**ZEB** - Road condition measures	Received upon request. Information in German.	data_in → NASA_SRTM30_tiles ➢ ZEB2017 ➢ ZEB2018
	Notes: the data set was provided by the German Federal Ministry for Digital and Transport, under an individual data usage agreement, prohibiting me from sharing it. There is no standardized way of obtaining this data. However, anyone can request it from BASt. As one can argue that this data falls under the German freedom of information act, everyone should be able to get access.		

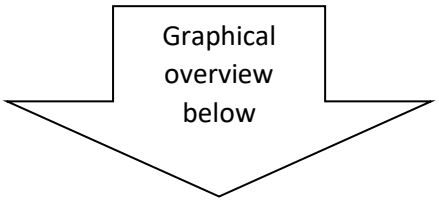
For details on institution names and data set licenses see: https://github.com/maikemp/motorwayData

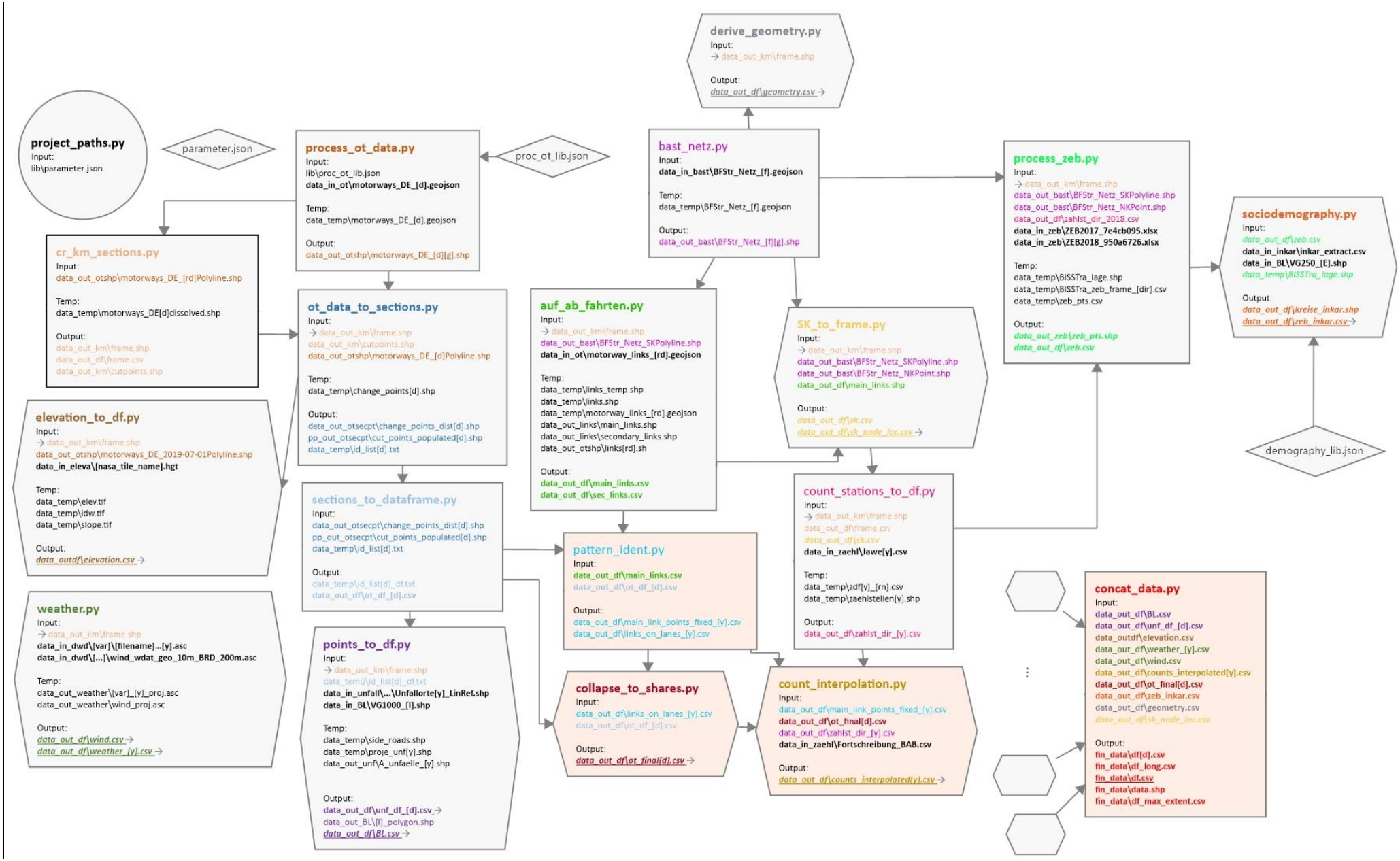
3. Setup programming environment

   Running these scripts requires access to the **arcpy** package, which comes with ArcGIS Pro.

4. Run files individually

   Note: due to the complexity of the project and the large number of involved data sources, for which we cannot rule out changes over time, the intermediate results should be checked at every step. Therefore, no main script or project running all code in a block is provided.




project_paths.py
Input:
lib\parameter.json
parameter.json
process_ot_data.py
Input:
lib\proc_ot_lib.json
data_in_ot\motorways_DE_[d].geojson
Temp:
data_temp\motorways_DE_[d].geojson
Output:
data_out_otshp\motorways_DE_[d][g].shp
proc_ot_lib.json
derive_geometry.py
Input:
→ data_out_km\frame.shp
Output:
data_out_df\geometry.csv →
bast_netz.py
Input:
data_in_bast\BFStr_Netz_[f].geojson
Temp:
data_temp\BFStr_Netz_[f].geojson
Output:
data_out_bast\BFStr_Netz_[f][g].shp
process_zeb.py
Input:
→ data_out_km\frame.shp
data_out_bast\BFStr_Netz_SKPolyline.shp
data_out_bast\BFStr_Netz_NKPoint.shp
data_out_df\zahlst_dir_2018.csv
data_in_zeb\ZEB2017_7e4cb095.xlsx
data_in_zeb\ZEB2018_950a6726.xlsx
Temp:
data_temp\BISSTra_lage.shp
data_temp\BISSTra_zeb_frame_[dir].csv
data_temp\zeb_pts.csv
Output:
data_out_zeb\zeb_pts.shp
data_out_df\zeb.csv
sociodemography.py
Input:
data_out_df\zeb.csv
data_in_inkar\inkar_extract.csv
data_in_BL\VG250_[E].shp
data_temp\BISSTra_lage.shp
Output:
data_out_df\kreise_inkar.shp
data_out_df\zeb_inkar.csv →
demography_lib.json
cr_km_sections.py
Input:
data_out_otshp\motorways_DE_[rd]Polyline.shp
Temp:
data_temp\motorways_DE[d]dissolved.shp
Output:
data_out_km\frame.shp
data_out_df\frame.csv
data_out_km\cutpoints.shp
ot_data_to_sections.py
Input:
→ data_out_km\frame.shp
data_out_km\cutpoints.shp
data_out_otshp\motorways_DE_[d]Polyline.shp
Temp:
data_temp\change_points[d].shp
Output:
data_out_otsecpt\change_points_dist[d].shp
pp_out_otsecpt\cut_points_populated[d].shp
data_temp\id_list[d].txt
auf_ab_fahrten.py
Input:
→ data_out_km\frame.shp
data_out_bast\BFStr_Netz_SKPolyline.shp
data_in_ot\motorway_links_[rd].geojson
Temp:
data_temp\links_temp.shp
data_temp\links.shp
data_temp\motorway_links_[rd].geojson
data_out_links\main_links.shp
data_out_links\secondary_links.shp
data_out_otshp\links[rd].sh
Output:
data_out_df\main_links.csv
data_out_df\sec_links.csv
SK_to_frame.py
Input:
→ data_out_km\frame.shp
data_out_bast\BFStr_Netz_SKPolyline.shp
data_out_bast\BFStr_Netz_NKPoint.shp
data_out_df\main_links.shp
Output:
data_out_df\sk.csv
data_out_df\sk_node_ioc.csv →
elevation_to_df.py
Input:
→ data_out_km\frame.shp
data_out_otshp\motorways_DE_2019-07-01Polyline.shp
data_in_eleva\[nasa_tile_name].hgt
Temp:
data_temp\elev.tif
data_temp\idw.tif
data_temp\slope.tif
Output:
data_outdf\elevation.csv →
sections_to_dataframe.py
Input:
data_out_otsecpt\change_points_dist[d].shp
pp_out_otsecpt\cut_points_populated[d].shp
data_temp\id_list[d].txt
Output:
data_temp\id_list[d]_df.txt
data_out_df\ot_df_[d].csv
pattern_ident.py
Input:
data_out_df\main_links.csv
data_out_df\ot_df_[d].csv
Output:
data_out_df\main_link_points_fixed_[y].csv
data_out_df\links_on_lanes_[y].csv
count_stations_to_df.py
Input:
→ data_out_km\frame.shp
data_out_df\frame.csv
data_out_df\sk.csv
data_in_zaehl\Jawe[y].csv
Temp:
data_temp\zdf[y]_[rn].csv
data_temp\zaehlstellen[y].shp
Output:
data_out_df\zahlst_dir_[y].csv
weather.py
Input:
→ data_out_km\frame.shp
data_in_dwd\[var]\[filename]...[y].asc
data_in_dwd\[...]\wind_wdat_geo_10m_BRD_200m.asc
Temp:
data_out_weather\[var]_[y]_proj.asc
data_out_weather\wind_proj.asc
Output:
data_out_df\wind.csv →
data_out_df\weather_[y].csv →
points_to_df.py
Input:
→ data_out_km\frame.shp
data_temü\id_list[d]_df.txt
data_in_unfall\...\Unfallorte[y]_LinRef.shp
data_in_BL\VG1000_[l].shp
Temp:
data_temp\side_roads.shp
data_temp\proje_unf[y].shp
data_out_unf\A_unfaelle_[y].shp
Output:
data_out_df\unf_df_[d].csv →
data_out_BL\[l]_polygon.shp
data_out_df\BL.csv →
collapse_to_shares.py
Input:
data_out_df\links_on_lanes_[y].csv
data_out_df\ot_df_[d].csv
Output:
data_out_df\ot_final[d].csv →
count_interpolation.py
Input:
data_out_df\main_link_points_fixed_[y].csv
data_out_df\ot_final[d].csv
data_out_df\zahlst_dir_[y].csv
data_in_zaehl\Fortschreibung_BAB.csv
Output:
data_out_df\counts_interpolated[y].csv →
concat_data.py
Input:
data_out_df\BL.csv
data_out_df\unf_df_[d].csv
data_outdf\elevation.csv
data_out_df\weather_[y].csv
data_out_df\wind.csv
data_out_df\counts_interpolated[y].csv
data_out_df\ot_final[d].csv
data_out_df\zeb_inkar.csv
data_out_df\geometry.csv
data_out_df\sk_node_ioc.csv
Output:
fin_data\df[d].csv
fin_data\df_long.csv
fin_data\df.csv
fin_data\data.shp
fin_data\df_max_extent.csv