Argentina Transport Risk Analysis Documentation

Oxford Infrastructure Analytics

Sep 08, 2023

Contents

[1]

This documentation describes the datasets assembled and created, with the Python scripts that require implementation for the analysis of the Argentina Transport Risk Analysis (ATRA).

The modelling and analysis aims to support decision-making by identifying the performance of adaptation options under current and future scenarios. It comprises a network flow model, generation of failure scenarios, economic impact assessment, and cost-benefit analysis of adaptation options.

[1] https://github.com/oi-analytics/argentina-transport/

CHAPTER 1

Contents

1.1 Installation

```
conda env create -f environment.yml
conda install python-igraph
```

See http://igraph.org/python/ for instructions on Windows installation of `python-igraph`.

Activate the environment:

```
conda activate argentina-transport
```

Set up the `atra` package (this project) for development use:

```
python setup.py develop
```

1.1.1 GAMS

The macroeconomic loss model uses GAMS[2] (General Algebraic Modeling System) via its python API. GAMS provide installation and licensing[3] instructions.

Configuration

The location of data and output files are configured by a `config.json` file. To point the scripts to the shared folder locations:

- copy `config.template.json` to `config.json`
- edit `config.json` to provide the paths to your working copy of your system directories (scripts may assume that file locations within the shared folders are consistent)
  - `incoming_data`: /incoming_data
  - `data`: /data
  - `figures`: /figures
  - `output`: /results

[2] https://www.gams.com/
[3] https://www.gams.com/latest/docs/UG_MAIN.htm

Note that on Windows, you will need to use double backslashes (\\) in the file paths, for example:

```
"data": "C:\\Users\\Username\\projects\\atra\\data"
```

1.2 Organization of Data

Important:

- This section describes how the data for the Argentina Transport Risk Analysis (ATRA) is organized in folders
- To implement the ATRA without any changes in existing codes, all data described here should be stored exactly as indicated below

1.2.1 Input and Output folders

All data are organised within the folders:

- `incoming_data`: Contains data obtained from various organizations in Argentina, which requires cleaning and processing for setting up the model analysis
- `data`: Contains cleaned data that used in the model analysis
- `results`: Contains the results of the model analysis
- `figures`: Contains maps and graph outputs

Note:

- Changes made to contents of the `incoming_data` and `data` folders will affect the Python scripts
- If the users change the file and folder paths within the `incoming_data` and `data` folders then they will have to modify the Python script that need these files and folders as inputs
- All data in the `results` and `figures` folders can be recreated by implmenting the Python scripts

1.3 Required data inputs and paramters

Important:

- This section describes the required data inputs and parameters for the Argentina Transport Risk Analysis (ATRA)
- To implement the ATRA all data described here should be created with the data properties and column names as described below
- If these data properties and column names are not provided in the data then the Python scripts will give run-time errors

1.3.1 Spatial data requirements

1. **All spatial data inputs must:**
    - Be projected to a valid coordinate system. Spatial data with no projection system will give errors
    - Have valid geometries. Null or Invalid geometries will give errors

Note:

- The assumed projection system used in the model is EPSG:4326
- If the users change any spatial data they have to create new data with a valid projection system

1.3.2 Topological network requirements

1. A topological network is defined as a graph composed of nodes and edges
2. **All finalised networks data are created and stored:**
    - In the file path - `/data/network/`
    - As csv file with post-processed network nodes and edges
    - As Shapefiles with post-processed network nodes and edges
    - The created networks are: `road, rail, port, air`
    - The `air` network has fewer attributes due to lack of data and airlines not being important for commidity flows
    - `bridge` files are also created but they are not networks, as explained below

Note: The names and properties of the attributes listed below are the essential network parameters for the whole model analysis. If the users wish to replace or change these datasets then they must retain the same names of columns with same types of values as given in the original data.

It is recommended that changes in parameter values should be made in the csv files, while the Shapefiles are mainly used for associating the geometries of the features. While we have provided the Shapefiles with parameter values as well, the model uses the Shapeflies mainly for performing geometry operations.

For example if a new road edge is added to the road network, then all its properties should be added to the `road_edges.csv` file, while in the `road_edges.shp` file the `edge_id` and valid `geometry` should be added.

The essential attributes in these networks are listed below. See the data for all attributes and try to recreate your data with similar column names and attribute values.

Several of these parameters and their values are created from `incoming_data` which is explained in the section Preparing data for the model[4]

3. **All nodes have the following attributes:**
    - `node_id` - String Node ID
    - `geometry` - Point geometry of node with projection ESPG:4326
    - Several other atttributes depending upon the specific transport sector
4. **All edges have the following attributes:**
    - `from_node` - String node ID that should be present in node_id column
    - `to_node` - String node ID that should be present in node_id column
    - `edge_id` - String edge ID
    - `geometry` - LineString geometry of edge with projection ESPG:4326
    - `length` - Float estimated length in kilometers of edge
    - `min_speed` - Float estimated minimum speed in km/hr on edge
    - `max_speed` - Float estimated maximum speed in km/hr on edge
    - `min_time` - Float estimated minimum time of travel in hours on edge

[4] https://argentina-transport-risk-analysis.readthedocs.io/en/latest/predata.html

- `max_time` - Float estimated maximum time of travel in hours on edge
- `min_gcost` - Float estimated minimum generalized cost in USD/ton on edge (not present in road edge files)
- `max_gcost` - Float estimated maximum generalized cost in USD/ton on edge (not present in road edge files)
- Several other atttributes depending upon the specific transport sector

---

Note: It is very important that the first 2 columns of the edge files should be `from_node` and `to_node`. If it is not then the network graph creation with give run-time error. The order of other columns is flexible. For example `edge_id` could be column number 3 or any other column after the second column.

---

5. **Attributes only present in roads edges:**
    - `road_name` - String name or number of road
    - `surface` - String value for surface material of the road
    - `road_type` - String value of either national, province or rural
    - `width` - Float width of edge in meters
    - `min_time_cost` - Float estimated minimum cost of time in USD on edge
    - `max_time_cost` - Float estimated maximum cost of time in USD on edge
    - `min_tariff_cost` - Float estimated minimum tariff cost in USD on edge
    - `max_tariff_cost` - Float estimated maximum tariff cost in USD on edge
    - `tmda_count` - Integer number of daily vehicle counts on edge
6. **National-roads bridges GIS data are also created as nodes containing:**
    - `bridge_id` - String bridge ID
    - `edge_id` - String edge ID matching `edge_id` of national-roads edges intersecting with bridges
    - `width` - Float with of bridge in meters
    - `length` - Float length of bridge in meters
    - `geometry` - Point geometry of node with projection ESPG:4326
    - Several other atttributes depending upon the specific bridge input data
7. **National-roads bridges GIS data are also created as edges containing:**
    - `bridge_id` - String bridge ID
    - `length` - Float length of bridge in meters
    - `geometry` - LineString geometry of bridge with projection ESPG:4326

Note: We assume that networks are provided as topologically correct connected graphs: each edge is a single LineString (may be straight line or more complex line), but must have exactly two endpoints, which are labelled as `from_node` and `to_node` (the values of these attributes must correspond to the `node_id` of a node).

Wherever two edges meet, we assume that there is a shared node, matching each of the intersecting edge endpoints. For example, at a t-junction there will be three edges meeting at one node.

Due to gaps in geometries and connectivity in the raw datasets several dummy nodes and edges have been created in the node and edges join points and lines. For example there are more nodes in the rail network than stations in Argentina, and similarly in the port network. The road network contains severral edges with `road_type = 0` which represent a dummy edge created to join two roads.

The bridge datasets are not networks because they do not have a topology. Bridge nodes are matched to the road network to later match road flow and failure results with failed bridges. For example, we estimate the failure

consequence of a road edge of the National Route 12 first, and if we know there is a bridge on this road that is also flooded then we assign the failure consequence to the bridge as well. Bridge edges are created to intersect with flood outlines to estimate the length of flooding of bridges.

1.3.3 OD matrices requirements

1. **All finalised OD matrices are stored:**
    - In the path - `/data/OD_data/`
    - As csv file with names `{mode}_nodes_daily_ods.csv` where `mode = {road, rail, port}`
    - As csv file with names `{mode}_province_annual_ods.csv`
    - As Excel sheets with combined Province level annual OD matrices
2. **All node-level daily OD matrices contain mode-wise and total OD flows and should have attributes:**
    - `origin_id` - String node IDs of origin nodes. Value should be present in the `node_id` column of the sectors network file
    - `destination_id` - String node IDs of destination nodes. Value should be present in the `node_id` column of the sectors network file
    - `origin_province` - String names of origin Provinces
    - `destination_province` - String names of destination Provinces
    - `min_total_tons` - Float values of minimum daily tonnages between OD nodes
    - `max_total_tons` - Float values of maximum daily tonnages between OD nodes
    - Float values of daily min-max tonnages of commodities/industries between OD nodes: here based on OD data provided for each sector
    - If min-max values cannot be estimated then there is a `total_tons` column - for roads only
3. **All aggregated province-level OD matrices contain mode-wise and total OD flows and should have attributes:**
    - `origin_province` - String names of origin Provinces
    - `destination_province` - String names of destination Provinces
    - `min_total_tons` - Float values of minimum daily tonnages between OD Provinces
    - `max_total_tons` - Float values of maximum daily tonnages between OD Provinces
    - Float values of daily min-max tonnages of commodities/industries between OD Provinces: here based on OD data provided for each sector
    - If min-max values cannot be estimated then there is a `total_tons` column - for roads only

Note: The OD columns names and their attributes listed aobve are essential for the flow and failure model analysis. While the names of commodities/industries might vary it is important that the OD data has the columns specifically mentioned as `origin_id`, `destination_id`, `origin_province`, `destination_province`, `min_total_tons (or total_tons)`, `max_total_tons (or total_tons)`.

The model can track individual commodity/industry flows and failure results, but in the overrall calculations it estimates the flows and disruptions corresponding to the total tonnage (min or max). The commodity/industry names are important for doing macroeconomic loss analysis explained below.

Hence, if an new user input contains only the total tonnage values and no commodity/industry specific OD values, then the model codes will still run with no errors, except the macroeconomic analysis code will not be able to run.

If the users wish to replace or change these datasets then they must retain the same names of columns with same types of values as given in the original data.

1.3.4 Hazards data requirements

1. **All hazard datasets are stored:**
    - In sub-folders in the path - `/data/flood_data/FATHOM`
    - As GeoTiff files
    - See `/data/flood_data/hazard_data_folder_data_info.xlsx` for details of all hazard files
2. **Single-band GeoTiff hazard raster files should have attributes:**
    - values - between 0 and 1000 for flood depth in meters
    - raster grid geometry
    - projection systems: Default assumed = EPSG:4326

Note: The hazard datasets were obtained from a third-party consultant https://www.fathom.global who generated flood maps specific to this project

It is assumed that all hazard data is provided in GeoTiff format with a projection system. If the users want to introduce new hazard data then it should be in GeoTiff format only.

When new hazard files are given the `hazard_data_folder_data_info.xlsx` should be updated accordingly

1.3.5 Administrative areas with statistics data requirements

1. **Argentina boundary datasets are stored:**
    - In the path - `/incoming_data/admin_boundaries_and_census/departamento/`
    - In the path - `/incoming_data/admin_boundaries_and_census/provincia/`
    - As Shapefiles
2. **Global boundary dataset for map plotting are stored:**
    - In the path - `/data/boundaries/`
    - As Shapefiles
3. **Census boundary data are stored:**
    - In the path - `/incoming_data/admin_boundaries_and_census/radios censales/`
    - As a Shapefile

Note: The admin and boundary datasets were obtained from different sources in Argentina

Table 1: List of admin and boundary datasets obtained different resources in Argentina

Admin boundary	Source
Department	Provided through World Bank
Province	Provided through World Bank
All admin levels	https://www.naturalearthdata.com/downloads/10m-physical-vectors/
Census - 2010	https://www.indec.gov.ar/

Admin boundary layers are generally available online. For example at https://data.humdata.org/dataset/argentina-administrative-level-0-boundaries.

The department, province and census datasets are used in the model, while the global boundaries are mainly used for generaing map backgrounds

The names and properties of the attributes listed below are the essential boundary parameters for the whole model analysis. If the users wish to replace or change these datasets then they must retain the same names of columns with same types of values as given in the original data.

For example if a new census dataset is introduced then it should contain the column `poblacion` with new population numbers. The census data used here is at Department level, but it could be replaced with other boundary level census estimates as well.

---

4. **All Argentina Department boundary datasets should have the attributes:**
    - `name` - String names Spanish - attribute name changed to `department_name`
    - `OBJECTID` - Integer IDs - attribute name changed to `department_id`
    - `geometry` - Polygon geometries of boundary with projection ESPG:4326
5. **All Argentina Province boundary datasets should have attributes:**
    - `nombre` - String names Spanish - attribute name changed to `province_name`
    - `OBJECTID` - Integer IDs - attribute name changed to `province_id`
    - `geometry` - Polygon geometries of boundary with projection ESPG:4326
6. **All global boundary datasets should have attributes:**
    - `name` - String names of boundaries in English
    - `geometry` - Polygon geometry of boundary with projection ESPG:4326
7. **The census datasets should have attributes:**
    - `poblacion` - Float value of population
    - `geometry` - Polygon geometry of boundary with projection ESPG:4326

1.3.6 Macroeconomic data requirements

1. For the macroeconomic analysis first a multi-regional IO matrix for 24 provinces in Argentina is created from a national-level IO matrix and province level Gross Production Values (GPV) of IO Industries

2. The multi-regional macroeconoic IO data is created from data downloaded from the Instituto Nacional de Estadística y Censos (INDEC) website. The data is stored as:

- Industry and Commodity level IO accounts in the file path `data/economic_IO_tables/input/sh_cou_06_16.xls`
- Industry level GPV in the file path `data/economic_IO_tables/input/PIB_provincial_06_17.xls`
- Names of aggregated industries classification for Argentina in the file path `data/economic_IO_tables/input/industry_high_level_classification.xlsx`, which should be present in the IO and GPV data files

3. **A set of look-up tables are created to match commodities in the OD matrices to IO industries**
    - In the file in path - `data/economic_IO_tables/input/commodity_classifications-hp.xlsx`
    - The sheetnames in the excel file are `road, rail, port` corresponding to the sector for which OD matrices are created
    - `commodity_group` - String name of commodity group identified in the OD matrices data

- `commodity_subgroup` - String name of commodity subgroup identified in the OD matrices data
- `high_level_industry` - String name of aggregated industry present in the `industry_high_level_classification.xlsx` file

4. **The multi-regional macroeconomic IO data creation, explained later, produces results:**
    - In the file in path - `data/economic_IO_tables/output/IO_ARGENTINA.xlsx`
    - In the file in path - `data/economic_IO_tables/output/MRIO_ARGENTINA_FULL.xlsx`
    - This data is used in the macroeconomic loss analysis

Note: The macroeconomic data are obtained from INDEC at https://www.indec.gob.ar/nivel3_default.asp?id_tema_1=3&id_tema_2=9&fbclid=IwAR02qnMIJeu86xUM5TFK5hrABN3FcJLGx6k5BYNhxLe4o0FhqJxuV2wxb5E. The PIB and COU datasets are used in the model

If the users want to update the IO tables for Argentina then it is recommended that they replace the above files `sh_cou_06_16.xls` and `PIB_provincial_06_17.xls` with exactly the same sheetnames and data structures as given in the original data used by the IO model scripts.

If the industry classifications are modified in the IO data then the changeas should also be made in `industry_high_level_classification.xlsx` and `commodity_classifications-hp.xlsx` files.

1.3.7 Adaptation options and costs requirements

1. **All adaptation options input datasets are stored:**
    - In the file - `/data/adaptation_options/ROCKS - Database - ARNG (Version 2.3) Feb2018.xlsx`
    - We use the sheet `Resultados Consolidados` for our analysis

Note: The adaptation data is very specific and if new options are created then the users will need to change the scripts as well

---

1.4 Pre-processing data for the model

Important: The topological network data and parameters described in Topological network requirements[5] had to be created from several data sources, which had several gaps.

- This section describes collected datasets that are used to create data for the Argentina Transport Risk Analysis (ATRA)
- The datasets listed here are specfic to Argentina and are used as inputs to make the finalized data used in the rest of the model
- To implement the ATRA pre-processing without any changes in existing codes, all data described here should be created and stored exactly as indicated below
- Python scripts were created specific to clean and modify these datasets, due to which changes made to the way the data are organized will most probably also result in making changes to the Python scripts
- In some instances some values for data are encoded within the Python scripts, so the users should be able to make changes directly in the Python scripts
- In each Python script described below, see the inline comments to understand where the inputs are given

[5] https://argentina-transport-risk-analysis.readthedocs.io/en/latest/parameters.html#topological-network-requirements

- It is recommended to run the Python scripts in the same order as described here
- If the users want to use the same data and make modifications in values of data then they can follow the steps and codes explained below. Otherwise this whole process can be skipped if the users know how to create the networks in the formats specified in the Topological network requirements[6]
- If the data are updated, especially if OD flows are updated to another year, then the users will have to make changes to the Python codes to be able to input new data files
- Mostly all inputs are read using the Python libraries of pandas[7] and geopandas[8]. The user should familiarise themselves with file reading and writing functions in these libraries. For example most codes use the geopandas function read_file[9] and to_file <http://geopandas.org/io.html>'_to read and write shapefiles, and the pandas functions 'read_excel and to_excel[10] and read_csv and to_csv[11] to read and write excel and csv data respectively

1.4.1 Creating the road network

Note: The road network is combined from datasets of national, provincial and rural roads in Argentina. The raw GIS data for these three types of networks were obtained from the Ministry of Transport and Dirección Nacional de Vialidad (DNV)

Table 2: List of road datasets obtained from different resources in Argentina

Road data	Source
National	https://www.argentina.gob.ar/vialidad-nacional/sig-vial
Province	Provided through World Bank from MoT
Rural	Provided through World Bank from MoT
National roads bridges	https://www.argentina.gob.ar/vialidad-nacional/sig-vial
OpenStreetMaps (OSM)	https://openmaptiles.com/downloads/dataset/osm/south-america/argentina/#2.96/-40.83/-63.6
National roads widths	Provided through World Bank from DNV
National roads speeds	Provided through World Bank from DNV
Road vehicle costs	Provided through World Bank from DNV

The portal https://ide.transporte.gob.ar/geoserver/web/ also contains open-source transport data that was downloaded, including the province and rural road networks. See the Python script `atra.preprocess.scrape_wfs`

1. **The road network data is stored:**
    - In sub-folders in the file path - `/data/pre_processed_networks_data/roads/`
    - As Shapefiles with attributes
    - **File in sub-folder `/national_roads/rutas/` contains national roads**
        - We extract the columns `cod_ruta` (for `road_name`), `sentido = A` and `geometry`
    - **File in sub-folder `/province_roads/` contains province roads**
        - We extract the columns `nombre` (for `road_name`), `clase` (for `surface`) and `geometry`

[6] https://argentina-transport-risk-analysis.readthedocs.io/en/latest/parameters.html#topological-network-requirements
[7] https://pandas.pydata.org
[8] http://geopandas.org
[9] http://geopandas.org/io.html
[10] http://pandas.pydata.org/pandas-docs/stable/user_guide/io.html
[11] http://pandas.pydata.org/pandas-docs/stable/user_guide/io.html

- **File in sub-folder `/rural_roads/` contains rural roads**
  - We extract the columns `characteris` (for `surface`) and `geometry`
- **File in sub-folder `/osm_roads/` contains OSM roads for gap filling**
  - We extract the columns `road_name`, `road_type` and `geometry`

2. **National Roads specifc GIS data are stored:**
   - In sub-folders in the path - `/incoming_data/pre_processed_network_data/roads/national_roads/`
   - As Shapefiles with attributes
   - **File in sub-folder `/indice_de_estado/` contains road surface quality as numeric values**
     - We use the columns `nro_regist` as id,‘‘valor‘‘ for `road_quality`, `sentido = A` and `geometry`
     - Road surface quality is used to estimate speeds on the national roads
   - **File in sub-folder `/indice_de_serviciabilidad/` contains road service quality as numeric values**
     - We use the columns `nro_regist` as id,‘‘valor‘‘ for `road_service`, `sentido = A` and `geometry`
     - Road service quality is used to estimate speeds on the national roads
   - **File in sub-folder `/materialcarril_sel/` contains road surface meterial as string values**
     - We use the columns `id_materia` as id,‘‘grupo‘‘ for `material_group`,‘‘sentido = A‘‘ and `geometry`
     - Surface material determines the conditon of the national roads for adaptation investments
   - **File in sub-folder `/tmda/` contains TMDA counts as numeric values**
     - We use the columns `nro_regist` as id,‘‘valor‘‘ for `road_service`, `sentido = A` and `geometry`
     - TMDA gives observed vehcile counts on national roads
   - **File in sub-folder `/v_mojon/` contains locations of kilometer markers**
     - We use the columns `id`, `progresiva`, `distancia` and `geometry`
     - kilometer markers are used in assinging properties on national roads and locating bridges
3. **Data on select national roads widths and terrains are stored:**
   - In the Excel file path - `incoming_data/road_properties/Tramos por Rutas.xls`
   - We use the sheet `Hoja1`
4. **Data on select national roads speeds are stored:**
   - In the Excel file path - `incoming_data/road_properties/TMDA y Clasificacion 2016.xlsx`
   - We use the sheet `Clasificación 2016`
5. **Road costs are stored:**
   - In the path - `/incoming_data/costs/road/`
   - As Excel files
   - The Vehicle Operating Costs are in the file `Costos de Operacion de Vehiculos.xlsx`

- We use the sheet `Camión Pesado` for costs
- The tariff costs are in the file `tariff_costs.xlsx`

Note:

The finalized road network is created by executing 3 Python scripts:

- Run *`atra.preprocess.combine_roads`* to extract data from the files described in Step 1 above
- Run `atra.preprocess.network_road_topology` to create road nodes and edges topology
- Run `atra.preprocess.road_network_creation` to assign road properties described above. This is the main script that creates the finalized road network and requires several inputs

The result of these scripts create the `road_edges` and `road_nodes` files described in the folder path `data/network/`

The topology script above is very specific to the case of the particular input data provide here. Unfortunaly if the data is changed them the users might have to test their results again if they run the topology script. We had to manually clean, edit and add some new edges to complete the topology. But this depends upon the quality of input provided and not the python script!

The Python codes require the specific inputs of the above datasets from the users to be able to identify the specific rows and columns in the data. If the users change these datasets in the future then, to use the same Python codes, then should preserve the column names and their properties

In the excel sheets in `incoming_data/road_properties/` and `incoming_data/costs/road/` the original data obtained from the DNV are preserved, and changing the locations and columns and rows will require making changes to the scripts. When data is missing some assumptions of values are taken, which are hard coded in the Python script.

The users should familiarize themselves with the functions in the script `atra.preprocess.road_network_creation` if they want to change data. Below the kinds of user inputs changes in this script are explained

- Lines 445-554 where all the inputs are given to the code. See the function:py:mod:*main*
- Currency exchange rate from ARS to USD is 1 ARS = 0.026 USD. See the function:py:mod:*main*
- The default `surface` of a national road is assumed to be `Asfalto`, and other roads it is `Tierra`. See the function `assign_road_surface`
- The default `width` of national and province roads is assumed to be 7.3m (2-lane) and rural roads is 3.65m (1-lane). The default `terrain` is assumed flat. See the function `assign_road_terrain_and_width`
- If no informattion on road speeds is provided through the data in `incoming_data/road_properties/TMDA y Clasificacion 2016.xlsx` then the road speeds are assumed to be as following. See the function `assign_min_max_speeds_to_roads`
- For national roads with poor to fair quality (0 < `road_service` <= 1) or (0 < `road_quality` <= 3) speeds vary from 50-80 km/hr
- For national roads with fair to good quality (1 < `road_service` <= 2) or (3 < `road_quality` <= 6) speeds vary from 60-90 km/hr
- For national roads with good to very good quality speeds vary from 70-100 km/hr
- For all province roads speeds vary from 40-60 km/hr
- For all rural roads speeds vary from 20-40 km/hr

1.4.2 Creating the national roads bridges data

1. **National-roads bridges GIS data are stored:**

- In the path - `/incoming_data/pre_processed_network_data/bridges/puente_sel/`
- As Shapefiles with Point geometry of nodes with projection ESPG:4326
- As Excel file with bridges attributes in sheetname `Consulta`

Note:

The finalized national-roads bridges data is created by executing 1 Python script after the road network has been already cr

- Run `atra.preprocess.road_bridge_matches` to extract data from the files described in Step 1 above

The original bridges data downloaded from https://www.argentina.gob.ar/vialidad-nacional/sig-vial provided a shapefile with only bridge locations, and the excel sheet with bridge properties. Unfortunately these two files did not have a common ID column to link them together. Hence the python script mainly matches the bridges to their location information using the kilometer marker locations specified for the bridge Excel data and matching these with the kilometer markers and national roads GIS data provided for the national roads, explained in Creating the road network[12]. If the users alrready have a bridge dataset has all attribtues in a geocoded files, then they do not need to run the Python script. But they will still have to match the `bridge_id` to the `edge_id` column of the `road_edges` dataset.

The result of this script creates the `bridge_edges` and `bridges` files described in the folder path `data/network/`. If th

- `id_estruct` - Numeric values to ID column only present in shapefile
- `ids` - Numeric values of bridge ID. Renamed to `bridge_id` by the model
- `longitud` - Float values of bridge length in meters. Renamed to `length` by the model
- `ancho de vereda derecha` - Float values of right lane width of bridge in meters. Used for estimating `width`
- `ancho de vereda izquierda` - Float values of left lane width of bridge in meters. Used for estimating `width`
- `ancho pavimento asc.` - Float values of pavement width of bridge in meters. Used for estimating `width`
- `ancho pavimento desc.` - Float values of pavement width of bridge in meters. Used for estimating `width`
- `tipo de estructura` - String description of the type of bridge. Renamed to `structure_type` by the model
- `ruta` - String name to national road where bridge belongs
- `geometry` - Point and line geometries of bridges with projection ESPG:4326
- Several other attributes which are not used in the rest of the model

---

1.4.3 Creating road OD matrix at node level

Note:

The road OD matrix data is matched to the `road_nodes` data by executing 1 Python script after the road network has bee

- Run `atra.preprocess.road_od_flows` to create the road OD matrix at node-node level

[12] https://argentina-transport-risk-analysis.readthedocs.io/en/latest/predata.html#creating-the-road-network

The original road OD data provided by the Secretaria de Planificacion de Cargas contains high-level annual OD matrices fo

- The nodes on national and province roads are only considered as OD nodes
- For each node the near population (obtained from census data) is estimated and only those nodes with population above 1000 are considered as OD nodes
- The OD nodes flows allocation is similar to a gravity model based on the importance of origin and destination nodes in creating and attracting OD flows.
- The OD matrices are annual and are converted to daily flows by dividing by 365

If the users want to change the high-level OD data then they should replace the OD datasets as described below. They can also can update the `road_nodes`, province and census shapefiles described in Administrative areas with statistics data requirements[13]

---

1. **Road commodity OD matrices data are stored:**
    - In the path - `/incoming_data/OD_data/road/Matrices OD 2014- tablas/`
    - As Excel files
    - The name of the excel file and excel sheet correspond to commodity groups and subgroups
    - Each Excel Sheet is a 123-by-123 matrix of OD tons with first row and first column showing Zone IDs
    - We use the sheets `Total Toneladas 2014` if given otherwise add tons across sheets
    - Each Excel Sheet is a 123-by-123 matrix with first row and first column showing Zone IDs
2. **Road commodity OD Zone data is stored:**
    - In the path - `/incoming_data/OD_data/road/Lineas de deseo OD- 2014/3.6.1.10.zonas/`
    - As Shapefile
    - `data` - The `od_id` that matches the OD matrices Excel data
    - `geometry` - Polygon geometry of zone with projection ESPG:4326

1.4.4 Creating the rail network and OD matrix

Note:

The finalized rail network and OD matrix data are all created by executing 1 Python script:

- Run `atra.preprocess.rail_od_flows` to create the rail network and OD matrix at node-node level

Table 3: List of rail datasets obtained from different resources in Argentina

Rail data	Source
Rail lines	Provided through World Bank from MoT
Stations	Provided through World Bank from MoT
OD data	Secretaria de Planificacion de Cargas
Transport Costs	Estimated from COSFER model by Secretaria de Planificacion de Transporte

Rail GIS data can also be downloaded from the portal https://ide.transporte.gob.ar/geoserver/web/. See the Python script `atra.preprocess.scrape_wfs`

[13] https://argentina-transport-risk-analysis.readthedocs.io/en/latest/parameters.html#administrative-areas-with-statistics-data-requirements

The original rail OD data provided by the Secretaria de Planificacion de Cargas contains station-station OD matrices which

- The names of the OD stations do not always match the nodes in the GIS data. So we do not always know the location of OD nodes
- The route information does not match any GIS data, if it exists
- In several cases the time-stamps are missing, so we do not know the time of start and end of a jounrey
- In several cases the distance of travel is missing, so we do not know the length of the jounrey
- Only is some instances does the data indicate the origin and destination provinces
- The GIS network shows several historic lines, which are no longer used. The GIS data does not indicate which lines are no longer in operation

The script `atra.preprocess.rail_od_flows` resolves some of the issues above. The following operations are perform

- The OD nodes are matched to GIS nodes
- The OD flows are routed on the GIS network, to check as best whether the observed OD distances match the estimated OD distances obtained from the GIS network. This helps in validating whether OD nodes were assigned correctly on the GIS network
- The total OD tonnages are aggregated over a day, based on the start date. From this the minimum and maximum OD flows are estimated betwork OD pairs
- Speeds are assigned based on the time-stamps of origin and destination stations. Default speeds of rail lines are assumed to be 20 km/hr

Unfortunalety the script `atra.preprocess.rail_od_flows` is very specific to the input datasets, and relies on having the same column names and organisation of data as described in the input data used in this current version

---

1. **Rail GIS data are stored:**
   - In the path - `/incoming_data/pre_processed_network_data/railways/national_rail/`
   - As Shapefiles

Note: The topology is assumed to have already been created in the rail network. We had to create some of this manually, so we cannot provide a automated Python script to do so. The user is recommended to check tools in the Python library snkit[14] for creating network topology.

---

2. **Rail OD matrices data are stored:**
   - In the path - `/incoming_data/OD_data/rail/Matrices OD FFCC/`
   - As Excel files
   - The names of the sheets within the excel files vary. See the Python script for specific information
   - The OD data in each excel sheet varies, but some information is necessary for OD matrix creation
   - `origin_station` - String name of origin station
   - `origin_date` - Datetime object for date of journey
   - `destination_station` - String name of destination station
   - `commodity_group` - String name of commodity groups
   - `line_name` - String name of thee line used for transport

[14] https://github.com/tomalrussell/snkit

- `tons` - Numeric values of tonnages
- Several other column, which are referred to in the Python script

3. **A file to match names of OD stations to GIS nodes is stored:**
    - In the path - `/incoming_data/pre_processed_network_data/railways/rail_data_cleaning/station_renames.xlsx`
    - As Excel file
    - This was created manually by looking at the OD and GIS data, and inferring matches based on Google searches and our judgement
4. **Rail costs are stored:**
    - In the Excel file path - `incoming_data/costs/rail/rail_costs.xlsx`
    - We use the sheet `route_costs`

1.4.5 Creating the port network and OD matrix

Note:

The port network and OD matrix data are all created by executing 1 Python script:

- Run `atra.preprocess.port_od_flows` to create the port network and OD matrix at node-node level

Table 4: List of port datasets obtained from different resources in Argentina

Port data	Source
Port locations	Secretaria de Planificacion de Cargas
Maritime routes	Created manually from OSM data
OD data	Secretaria de Planificacion de Cargas
Transport Costs	Estimated from data from Secretaria de Planificacion de Transporte

Port GIS node data can also be downloaded from the portal https://ide.transporte.gob.ar/geoserver/web/. See the Python script `atra.preprocess.scrape_wfs`

The original port OD data provided by the Secretaria de Planificacion de Cargas contains port specific OD data which are t

- The original data gives port specific information on how much different types of freight are exported, imported or transiting at the port
- The information on the origin and destination of the freights are mostly missing, so we have inferred them as best
- In several cases the time-stamps are missing, so we do not know the time of start and end of a jounrey
- Only is some instances does the data indicate the origin and destination provinces or countries

The script `atra.preprocess.port_od_flows` resolves some of the issues above. The following operations are perform

- The OD nodes are inferred by gap filling the port-level flow data
- The total OD tonnages are aggregated over a day, based on the start date. From this the minimum and maximum OD flows are estimated betwork OD pairs
- Default speeds are assumed to be 4-5 km/hr

Unfortunalety the script `atra.preprocess.port_od_flows` is very specific to the input datasets, and relies on having the same column names and organisation of data as described in the input data used in this current version

1. **Port GIS data are stored:**
    - In the path - `/incoming_data/pre_processed_network_data/ports/`
    - As Shapefiles

Note: The topology is assumed to have already been created in the rail network. We had to create some of this manually, so we cannot provide a automated Python script to do so. The user is recommended to check tools in the Python library snkit[15] for creating network topology.

---

2. **A file to match names of ports and commodity to GIS nodes is stored:**
    - In the path - `/incoming_data/pre_processed_network_data/ports/rail_od_cleaning/od_port_matches.xlsx`
    - As Excel file
    - This was created manually by looking at the OD and GIS data, and inferring matches based on Google searches and our judgement
3. **Port specific freight data are stored:**
    - In the Excel file path - `/incoming_data/OD_data/ports/Puertos/Cargas No Containerizadas - SSPVNYMM.xlsx`
    - We use the excel sheet `2017`
    - Some information is necessary for OD matrix creation
    - `Puerto` - String name of port where data is recorded
    - `Puerto de Procedencia` - String name of origin port
    - `País de Procedencia` - String name of origin country
    - `Fecha Entrada` - Datetime object for entrance date recorded at port
    - `Puerto de Destino` - String name of destination port
    - `País de Destino` - String name of destination country
    - `Producto Corregido` - String name of commodity subgroups
    - `Rubro` - String name of commodity groups
    - `Tipo de Operación` - String name of operation type, associated to exports, imports, and transit
    - `Total Tn` - Numeric values of tonnages
    - `Medida` - String value of type of tonnages
4. **Port costs are stored:**
    - In the Excel file path - `incoming_data/costs/port/port_costs.xlsx`
    - We use the excel sheet `costs`

1.4.6 Creating the air network and passenger data

Note:

The air network and passenger flow data are all created by executing 1 Python script:

- Run *`atra.preprocess.network_air`* to create the air network and passenger flows at node-node level

[15] https://github.com/tomalrussell/snkit

Table 5: List of port datasets obtained from different resources in Argentina

Air data	Source
Airport locations	https://ide.transporte.gob.ar/geoserver/web/
Passenger number - 2016	Secretaria de Planificacion de Cargas

Airport GIS nodee data is downloaded from the portal https://ide.transporte.gob.ar/geoserver/web/. See the Python script `atra.preprocess.scrape_wfs`

1. **Air passenger OD data is contained in the airlines shapefile**
    - In the file - `/data/pre_processed_networks_data/air/SIAC2016pax.shp`
    - Some information is necessary for OD matrix creation
    - `Cod_Orig` - String IATA code of origin airport
    - `Cod_Destt` - String IATA code of destination airport
    - `Pax_2016` - Numeric values of passenger numbers

1.4.7 Creating the multi-modal network edges

Note:

The multi-modal network edges are all created by executing 1 Python script:

- Run `atra.preprocess.multi_modal_network_creation`

The multi-modal edges can only be created once all the other network are created. The code inputs the finalized `road`, `rail` and `port` files in the `data/network/` folder path

1.4.8 Industry specific province-level OD matrix

Note:

For macroeconomic analysis an industry specific province-level OD matrix is created by executing 1 Python script:

- Run `atra.preprocess.od_combine`

The province OD matric can only be created once all the other OD matrices are created. The code inputs the finalized `{mode}_province_annual_ods.csv` OD files in the `data/OD_data/` folder path

1.4.9 Preparing Hazard Data

Note:

- **Convert GeoTiff raster hazard datasets to shapefiles based on flood depth thresholds**
    - Run *`atra.preprocess.convert_hazard_data`*
- Load data as described in Hazards data requirements[16]
- **Create hazard shapefiles with:**
    - `ID` - equal to 1
    - `geometry` - Polygon outline of selected hazard

[16] https://argentina-transport-risk-analysis.readthedocs.io/en/latest/parameters.html#hazards-data-requirements

- Store outputs in same paths in directory `/data/flood_data/FATHOM/`

1.5 Analysis and Results

Important:

- This section describes the steps Analysis and Results steps of the Argentina Transport Risk Analysis (ATRA)
- To implement the ATRA without any changes in existing codes, all data described here should be created and stored exactly as indicated below

1.5.1 Mapping Flows onto Networks

Purpose:

- **Map the national-scale OD node level matrix values to network paths**
  - For all transport modes at national scale
  - Estimate 2 values - A MIN and a MAX value of flows between each selected OD node pair
  - Based on MIN-MAX generalised costs estimates

Execution:

- Load data as described in Topological network requirements[17] and OD matrices requirements[18]
- For road, rail, port OD matrices run *`atra.analysis.flow_mapping`*

Result:

- Store OD flow paths in csv outputs in `/results/flow_mapping_paths/`
- Store total OD flows on edges in csv files in `/results/flow_mapping_combined/`
- Optional - Store OD flows on edges in shapefiles in `/results/flow_mapping_shapefiles/`
- **csv files in `/results/flow_mapping_paths/` contain attributes:**
  - `origin_id` - String node ID of Origin
  - `destination_id` - String node ID of Destination
  - `origin_province` - String name of Province of Origin node ID
  - `destination_province` - String name of Province of Destination node ID
  - `min_edge_path` - List of string of edge IDs for paths with minimum generalised cost flows
  - `max_edge_path` - List of string of edge IDs for paths with maximum generalised cost flows
  - `min_distance` - Float values of estimated distance for paths with minimum generalised cost flows
  - `max_distance` - Float values of estimated distance for paths with maximum generalised cost flows
  - `min_time` - Float values of estimated time for paths with minimum generalised cost flows
  - `max_time` - Float values of estimated time for paths with maximum generalised cost flows
  - `min_gcost` - Float values of estimated generalised cost for paths with minimum generalised cost flows

[17] https://argentina-transport-risk-analysis.readthedocs.io/en/latest/parameters.html#topological-network-requirements
[18] https://argentina-transport-risk-analysis.readthedocs.io/en/latest/parameters.html#od-matrices-requirements

  - `max_gcost` - Float values of estimated generalised cost for paths with maximum generalised cost flows
  - `min_total_tons` - Float values of estimated daily minimum total tonnages for all industries bettween OD pair
  - `max_total_tons` - Float values of estimated daily maximum total tonnages for all industries bettween OD pair
  - `industry_columns` - All daily tonnages of industry columns given in the OD matrix data for specific sectors
- **csv files in `/results/flow_mapping_combined/` contain attributes:**
  - `edge_id` - String edge ID
  - `min_total_tons` - Float values of estimated daily minimum total tonnages on edge
  - `max_total_tons` - Float values of estimated daily maximum total tonnages on edge
  - `commodity/industry_columns` - All total daily tonnages of commodity/industry columns on edge

1.5.2 Hazard Exposure

Purpose:

- **Intersect hazards and network line and point geometries with hazatd polygons**
  - Write final results to Shapefiles
- **Collect network-hazard intersection attributes**
  - Combine with boundary Polygons to collect network-hazard-boundary intersection attributes
  - Write final results to an Excel sheet

Execution:

- Load data as described in Topological network requirements[19] and Preparing Hazard Data[20], and Administrative areas with statistics data requirements[21]
- Run *`atra.analysis.hazards_networks_intersections`*
- Run *`atra.analysis.hazards_network_intersections_results_collect`*

Result:

- Store shapefile outputs in the directory `/results/networks_hazards_intersection_shapefiles/`
- **All hazard-edge intersection shapefiles with attributes:**
  - `edge_id` - String name of intersecting edge ID
  - `length` - Float length of intersection of edge LineString and hazard Polygon
  - `geometry` - LineString geometry of intersection of edge LineString and hazard Polygon
- **All hazard-node intersection shapefile with attributes:**
  - `node_id` - String name of intersecting node ID
  - `geometry` - Point geometry of intersecting node ID
- Store summarised results in csv files in path `/results/hazard_scenarios/`
- **csv files of network-hazard-boundary intersection with attributes:**
  - `edge_id/node_id` - String name of intersecting edge ID or node ID

[19] https://argentina-transport-risk-analysis.readthedocs.io/en/latest/parameters.html#topological-network-requirements
[20] https://argentina-transport-risk-analysis.readthedocs.io/en/latest/predata.html#preparing-hazard-data
[21] https://argentina-transport-risk-analysis.readthedocs.io/en/latest/parameters.html#administrative-areas-with-statistics-data-requirements

  - `length` - Float length of intersection of edge LineString and hazard Polygon: Only for edges
  - `province_id` - String/Integer ID of Province
  - `province_name` - String name of Province
  - `department_id` - String/Integer ID of Department
  - `department_name` - String name of Department
  - `hazard_type` - String name of hazard type
  - `model` - String name of hazard model
  - `year` - String name of hazard year
  - `climate_scenario` - String name of hazard scenario
  - `probability` - Float/String value of hazard probability
  - `min_depth` - Integer value of minimum value of flood depth of exposure
  - `max_depth` - Integer value of maximum value of flood depth of exposure

1.5.3 Combine hazard scenarios for risk weights

Purpose

- Combine failure scenarios across probability levels into single value per hazard type, scenario, network edges
- The risk weights are the sum of probability*exposure for each hazard type intersecting network edges

Execution

- Load results from Hazard exposure[22] and Topological network requirements[23]
- Run *`atra.analysis.collect_network_hazard_scenarios_national`*

Result

- Combined scenarios in `results/network_stats/{mode}_hazard_intersections_risk_weights.csv`
  - `edge_id/bridge_id` - string, name of failed edge
  - `hazard_type` - string, name of hazard
  - `model` - string, name of hazard model (if any)
  - `climate_scenario` - string, name of climate scenario (if any)
  - `year` - integer, year of hazard data
  - `edge_length` - float, length of edge
  - `min/max_height` - float, hazard height (if any)
  - `min/max_exposure_percent` - float, percentage of edge exposed to hazard
  - `min/max_duration_wt` - float, duration weight
  - `min/max_exposure_length` - float, length of edge exposed to hazard
  - `risk_wt` - float, risk weight
  - `dam_wt` - float, damage weight

[22] https://argentina-transport-risk-analysis.readthedocs.io/en/latest/results.html#hazard-exposure
[23] https://argentina-transport-risk-analysis.readthedocs.io/en/latest/parameters.html#topological-network-requirements

1.5.4 Network Failure Analysis

Purpose:

- **Failure analysis of edges in invidiual networks**
  - To estimate flow isolations and rerouting effects on same network
- **Failure analysis of edges in networks with multi-modal options**
  - To estimate flow isolations and rerouting effects with multi-modal options

Execution:

- Load network and flow excel data as described in Topological network requirements[24], Mapping Flows onto Networks[25], and failure scenarios from Hazard exposure[26]
- For all networks failure analysis run *atra.analysis.failure_estimation*
- For networks failure analysis with multi-modal options run *atra.analysis.multi_modal_failure_estimation*

Result:

- Store csv outputs in the directory `/results/failure_results/`
- Optional - Store shapefile outputs in `/results/failure_shapefiles/`
- **All failure scenarios results in `/results/failure_results/all_fail_scenarios/`**
  - `edge_id` - String name or list of failed edges
  - `origin_id` - String node ID of Origin of disrupted OD flow
  - `destination_id` - String node ID of Destination of disrupted OD flow
  - `origin_province` - String name of Province of Origin node ID of disrupted OD flow
  - `destination_province` - String name of Province of Destination node ID of disrupted OD flow
  - `no_access` - Boolean 1 (no reroutng) or 0 (rerouting)
  - `min/max_distance` - Float value of estimated distance of OD journey before disruption
  - `min/max_time` - Float value of estimated time of OD journey before disruption
  - `min/max_gcost` - Float value of estimated travel cost of OD journey before disruption
  - `new_cost` - Float value of estimated cost of OD journey after disruption
  - `new_distance` - Float value of estimated distance of OD journey after disruption
  - `new_path` - List of string edge IDs of estimated new route of OD journey after disruption
  - `new_time` - Float value of estimated time of OD journey after disruption
  - `dist_diff` - Float value of Post disruption minus per-disruption distance
  - `time_diff` - Float value Post disruption minus per-disruption timee
  - `min/max_tr_loss` - Float value of estimated change in rerouting cost
  - `industry_columns` - Float values of all daily tonnages of industry columns along disrupted OD pairs
  - `min/max_total_tons` - Float values of total daily tonnages along disrupted OD pairs
- **Isolated OD scenarios - OD flows with no rerouting options in `/results/failure_results/isolated_od`**

[24] https://argentina-transport-risk-analysis.readthedocs.io/en/latest/parameters.html#topological-network-requirements
[25] https://argentina-transport-risk-analysis.readthedocs.io/en/latest/results.html#mapping-flows-onto-networks
[26] https://argentina-transport-risk-analysis.readthedocs.io/en/latest/results.html#hazard-exposure

  - `edge_id` - String name or list of failed edges
  - `origin_province` - String name of Province of Origin node ID of disrupted OD flow
  - `destination_province` - String name of Province of Destination node ID of disrupted OD flow
  - `industry_columns` - Float values of all daily tonnages of industry columns along disrupted OD pairs
  - `min/max_total_tons` - Float values of total daily tonnages along disrupted OD pairs

- **Rerouting scenarios - OD flows with rerouting options in `/results/failure_results/rerouting_scen`**

  - `edge_id` - String name or list of failed edges
  - `o_region` - String name of Province of Origin node ID of disrupted OD flow
  - `d_region` - String name of Province of Destination node ID of disrupted OD flow
  - `min/max_tr_loss` - Float value of change in rerouting cost
  - `min/max_total_tons` - Float values of total daily tonnages along disrupted OD pairs

- **Min-max combined scenarios - Combined min-max results along each edge in `/results/failure_results`**

  - `edge_id` - String name or list of failed edges
  - `no_access` - Boolean 1 (no reroutng) or 0 (rerouting)
  - `min/max_tr_loss` - Float values of change in rerouting cost
  - `min/max_total_tons` - Float values of total daily tonnages affected by disrupted edge

- **Shapefile min-max combined scenarios**
  - `edge_id` - String name or list of failed edges
  - `no_access` - Boolean 1 (no reroutng) or 0 (rerouting)
  - `min/max_tr_loss` - Float values of change in rerouting cost
  - `min/max_total_tons` - Float values of total daily tonnages affted by disrupted edge
  - `geometry` - LineString geomtry of edges

1.5.5 Macroeconomic loss Analysis

Purpose:

- **Macroeconomic losses analysis due to edge failures in networks**
  - To estimate economic impacts of flow isolations/disruptions
  - To understand the wider economic impacts of these disruptions

Execution:

- Load data described in Macroeconomic Data[27] and OD matrices requirements[28]
- To create the multiregional input-output table for Argentina, run `atra.mrio.run_mrio`
- To perform the loss analysis, run `atra.mria.run_mria`

Result:

- **Store the new multiregional input-output table in `/data/economic_IO_tables/output_data/`**

[27] https://argentina-transport-risk-analysis.readthedocs.io/en/latest/parameters.html#macroeconomic-data-requirements
[28] https://argentina-transport-risk-analysis.readthedocs.io/en/latest/parameters.html#od-matrices-requirements

  - **files `IO_ARGENTINA.xlsx` contain:**
    - Sheetname `T` with the full multiregional table
    - Sheetname `labels_T` with the column and row labels of matrix `T`
    - Sheetname `FD` with the final demand columns of the new table
    - Sheetname `labels_FD` with the column labels of matrix `FD`
    - Sheetname `ExpROW` with the export to the Rest of the World columns of the new table
    - Sheetname `labels_ExpROW` with the column labels of matrix `ExpROW`
    - Sheetname `VA` with the value added rows of the new table
    - Sheetname `labels_VA` with the row labels of matrix `VA`
- Store csv files in `/results/economic_failure_losses/summarized/`
- **All summarized files have the following attributes:**
  - `edge_id` - String edge IDs
  - `total_losses` - Value of the total economic losses due to the disruption of the corresponding edge ID
- Store csv files in `/results/economic_failure_losses/od_region_losses/`
- **All od_losses file have the following attributes:**
  - `edge_id` - String edge IDs
  - `region` - String name of the region
  - `dir_losses` - Value of the direct losses due to the diruption of the corresponding edge ID in the corresponding region
  - `total_losses` - Value of the total losses due to the diruption of the corresponding edge ID in the corresponding region
  - `ind_losses` - Value of the indirect losses due to the diruption of the corresponding edge ID in the corresponding region

1.5.6 Combining Network Failure and Macroeconomic loss Results

Purpose:

- Combine macroeconomic loss estimates with rerouting losses

Execution:

- Load data described in Failure Analysis[29] and Macroeconomic loss analysis[30]
- Run *`atra.analysis.economic_failure_combine_national`*

Result:

- Store csv files in `/results/failure_results/minmax_combined_scenarios/`
- **Files with names `single_edge_failures_minmax_national_{mode}_{x}_percent_disrupt.csv`**
  - `edge_id` - String name or list of failed edges
  - `no_access` - Boolean 1 (no reroutng) or 0 (rerouting)
  - `min/max_tr_loss` - Float values of change in rerouting cost
  - `min/max_total_tons` - Float values of total daily tonnages affected by disrupted edge

[29] https://argentina-transport-risk-analysis.readthedocs.io/en/latest/results.html#failure-analysis
[30] https://argentina-transport-risk-analysis.readthedocs.io/en/latest/results.html#macroeconomic-loss-analysis

  - `min/max_econ_loss` - Float values of total daily macroeconomic losses
  - `min/max_econ_impact` - Float values of sum of transport loss and macroeconomic loss

1.5.7 Estimating the bridge flows and failure losses

Purpose:

- Estimate the flows and failure losses on the national-roads bridges
- This done after all road failure analysis is performed because bridges results are estimated through the road failures

Execution:

- Run *atra.analysis.failure_estimation_bridges*

Result:

- Creates outputs for bridges similar to the ones explained in Mapping Flows onto Networks[31]
- Creates outputs for bridges similar to the ones explained in Combining Network Failure and Macroeconomic loss Results[32]

1.5.8 Adaptation

Purpose:

- Generate adaption scenarios/strategies and examine their costs, benefits, net present values and benefit-cost ratios
- For roads and bridges, based on different types of hazards, road assets and climate-change conditions

Execution:

- Load data described in Topological network requirements[33], Combining Network Failure and Macroeconomic loss Results[34], and Adaptation Options[35]
- Common functions are in *atra.adaptation_options*
- Run *atra.analysis.adaptation_analysis*

Result:

- Store results as excel sheets in `/results/adaptation_results/`
- **All adaptation results have the following attributes:**
  - `edge_id/bridge_id` - string, edge or bridges IDs
  - `hazard_type` - string, names of hazard types
  - `model` - string, names of hazard models
  - `climate_scenario` - string, names of climate scenarios
  - `year` - integer, values of year of hazard climate models
  - `width` - float, edge widths
  - `edge_length` - float, edge lengths
  - `min/max_depth` - float, heights of hazard exposure - if flooding
  - `min/max_exposure_percent` - float, percent of edge length exposed to hazard
  - `min/max_duration_wt` - float, duration of disruption of edge

[31] https://argentina-transport-risk-analysis.readthedocs.io/en/latest/results.html#mapping-flows-onto-networks
[32] https://argentina-transport-risk-analysis.readthedocs.io/en/latest/results.html#combining-network-failure-and-macroeconomic-loss-results
[33] https://argentina-transport-risk-analysis.readthedocs.io/en/latest/parameters.html#topological-network-requirements
[34] https://argentina-transport-risk-analysis.readthedocs.io/en/latest/results.html#combining-network-failure-and-macroeconomic-loss-results
[35] https://argentina-transport-risk-analysis.readthedocs.io/en/latest/data.html#adaptation-options

- `min/max_exposure_length` - float, edge length exposed to hazard
- `risk_wt` - float, weight given to estimating expected annual losses
- `dam_wt` - float, weight given to estimating expected annual damage costs
- `min/max_econ_impact` - float, minimum/maximum economic impact
- `min/max_benefit` - float, minimum/maximum benefit
- `min/max_ini_adap_cost` - float, minimum/maximum initial adaptation cost
- `min/max_tot_adap_cost` - float, minimum/maximum total adaptation cost
- `min/max_bc_ratio` - float, minimum/maximum benefit cost ratio
- `min/max_bc_diff` - float, minimum/maximum benefit cost difference
- Attributes specific to the roads or bridges

1.5.9 Processing outputs and plots

Purpose:

- Several scripts are written to generate statistics and plots to process results
- These codes are very specific to the kinds of data and outputs produced from the analysis
- See the scripts with *`atra.stats`* and *`atra.plot`*

1.6 atra package

OI-Analytics utility package for Argentina project

1.6.1 Subpackages

atra.analysis package

Submodules

atra.analysis.adaptation_analysis module

Assess national adaptation options

`main`()

(i) estimated cost to upgrade to a climate-resilient bituminous 2L (applied to unpaved, gravel and bituminous 2L roads),

(ii) estimated cost to upgrade to a climate-resilient bituminous 4L

(applied to bituminous 4L roads),

(iii) estimated cost to upgrade to a climate-resilient concrete 2L

(applied to concrete 2L roads),

(iv) estimated cost to upgrade to a climate-resilient concrete 4L

(applied to concrete 4L roads).

atra.analysis.collect_network_hazard_scenarios_national module

Collect network hazard scenarios

main()
Process results

1. **Specify the paths from where you to read and write:**
    - Input data
    - Intermediate calcuations data
    - Output results
2. **Supply input data and parameters**
    - Names of the three Provinces - List of string types
    - Names of modes - List of strings
    - Names of output modes - List of strings
    - Names of hazard bands - List of integers
    - Names of hazard thresholds - List of integers
    - Condition 'Yes' or 'No' is the users wants to process results
3. **Give the paths to the input data files:**
    - Commune boundary and stats data shapefile
    - Hazard datasets description Excel file
    - String name of sheet in hazard datasets description Excel file

atra.analysis.economic_failure_combine_national module

Combine national-scale macroeconomic loss estimates with rerouting losses

correct_economic_loss_estimates(*x*)

main()
Process results

atra.analysis.failure_estimation module

Failure analysis of national-scale networks For transport modes at national scale:

- road
- rail

Input data requirements

1. Correct paths to all files and correct input parameters
2. csv sheets with results of flow mapping based on MIN-MAX generalised costs estimates:
    - origin - String node ID of Origin
    - destination - String node ID of Destination
    - origin_province - String name of Province of Origin node ID
    - destination_province - String name of Province of Destination node ID
    - min_edge_path - List of string of edge ID's for paths with minimum generalised cost flows
    - max_edge_path - List of string of edge ID's for paths with maximum generalised cost flows

- min_distance - Float values of estimated distance for paths with minimum generalised cost flows
- max_distance - Float values of estimated distance for paths with maximum generalised cost flows
- min_time - Float values of estimated time for paths with minimum generalised cost flows
- max_time - Float values of estimated time for paths with maximum generalised cost flows
- min_gcost - Float values of estimated generalised cost for paths with minimum generalised cost flows
- max_gcost - Float values of estimated generalised cost for paths with maximum generalised cost flows
- industry_columns - All daily tonnages of industry columns given in the OD matrix data

3. Shapefiles
    - edge_id - String/Integer/Float Edge ID
    - geometry - Shapely LineString geomtry of edges

Results

Csv sheets with results of failure analysis:

1. All failure scenarios
    - edge_id - String name or list of failed edges
    - origin - String node ID of Origin of disrupted OD flow
    - destination - String node ID of Destination of disrupted OD flow
    - origin_province - String name of Province of Origin node ID of disrupted OD flow
    - destination_province - String name of Province of Destination node ID of disrupted OD flow
    - no_access - Boolean 1 (no reroutng) or 0 (rerouting)
    - min/max_distance - Float value of estimated distance of OD journey before disruption
    - min/max_time - Float value of estimated time of OD journey before disruption
    - min/max_gcost - Float value of estimated travel cost of OD journey before disruption
    - new_cost - Float value of estimated cost of OD journey after disruption
    - new_distance - Float value of estimated distance of OD journey after disruption
    - new_path - List of string edge ID's of estimated new route of OD journey after disruption
    - new_time - Float value of estimated time of OD journey after disruption
    - dist_diff - Float value of Post disruption minus per-disruption distance
    - time_diff - Float value Post disruption minus per-disruption timee
    - min/max_tr_loss - Float value of estimated change in rerouting cost
    - min/max_tons - Float values of total daily tonnages along disrupted OD pairs
    - industry_columns - Float values of all daily tonnages of industry columns along disrupted OD pairs
2. Isolated OD scenarios - OD flows with no rerouting options
    - edge_id - String name or list of failed edges
    - origin_province - String name of Province of Origin node ID of disrupted OD flow
    - destination_province - String name of Province of Destination node ID of disrupted OD flow
    - industry_columns - Float values of all daily tonnages of industry columns along disrupted OD pairs
    - min/max_tons - Float values of total daily tonnages along disrupted OD pairs
3. Rerouting scenarios - OD flows with rerouting options

- edge_id - String name or list of failed edges
- origin_province - String name of Province of Origin node ID of disrupted OD flow
- destination_province - String name of Province of Destination node ID of disrupted OD flow
- min/max_tr_loss - Float value of change in rerouting cost
- min/max_tons - Float values of total daily tonnages along disrupted OD pairs

4. Min-max combined scenarios - Combined min-max results along each edge
   - edge_id - String name or list of failed edges
   - no_access - Boolean 1 (no reroutng) or 0 (rerouting)
   - min/max_tr_loss - Float values of change in rerouting cost
   - min/max_tons - Float values of total daily tonnages affted by disrupted edge

main()

Estimate failures

Specify the paths from where you want to read and write:

1. Input data
2. Intermediate calcuations data
3. Output results

Supply input data and parameters

1. **Names of modes** String
2. **Names of min-max tons columns in sector data** List of string types
3. **Min-max names of names of different types of attributes - paths, distance, time, cost, tons** List of string types
4. **Names of commodity/industry columns for which min-max tonnage column names already exist** List of string types
5. **Percentage of OD flows that are assumed disrupted** List of float type
6. **Condition on whether analysis is single failure or multiple failure** Boolean condition True or False

Give the paths to the input data files:

1. Network edges csv and shapefiles
2. OD flows csv file
3. Failure scenarios csv file

Specify the output files and paths to be created

atra.analysis.failure_estimation_bridges module

Failure analysis of national-scale networks For transport modes at national scale:

- road
- rail

Input data requirements

1. Correct paths to all files and correct input parameters
2. Excel sheets with results of flow mapping based on MIN-MAX generalised costs estimates:
   - origin - String node ID of Origin

- destination - String node ID of Destination
- o_region - String name of Province of Origin node ID
- d_region - String name of Province of Destination node ID
- min_edge_path - List of string of edge ID's for paths with minimum generalised cost flows
- max_edge_path - List of string of edge ID's for paths with maximum generalised cost flows
- min_distance - Float values of estimated distance for paths with minimum generalised cost flows
- max_distance - Float values of estimated distance for paths with maximum generalised cost flows
- min_time - Float values of estimated time for paths with minimum generalised cost flows
- max_time - Float values of estimated time for paths with maximum generalised cost flows
- min_gcost - Float values of estimated generalised cost for paths with minimum generalised cost flows
- max_gcost - Float values of estimated generalised cost for paths with maximum generalised cost flows
- min_vehicle_nums - Float values of estimated vehicle numbers for paths with minimum generalised cost flows
- max_vehicle_nums - Float values of estimated vehicle numbers for paths with maximum generalised cost flows
- industry_columns - All daily tonnages of industry columns given in the OD matrix data

3. Shapefiles
    - edge_id - String/Integer/Float Edge ID
    - geometry - Shapely LineString geomtry of edges

Results

Csv sheets with results of failure analysis:

1. All failure scenarios
    - edge_id - String name or list of failed edges
    - origin - String node ID of Origin of disrupted OD flow
    - destination - String node ID of Destination of disrupted OD flow
    - o_region - String name of Province of Origin node ID of disrupted OD flow
    - d_region - String name of Province of Destination node ID of disrupted OD flow
    - no_access - Boolean 1 (no reroutng) or 0 (rerouting)
    - min/max_distance - Float value of estimated distance of OD journey before disruption
    - min/max_time - Float value of estimated time of OD journey before disruption
    - min/max_gcost - Float value of estimated travel cost of OD journey before disruption
    - min/max_vehicle_nums - Float value of estimated vehicles of OD journey before disruption
    - new_cost - Float value of estimated cost of OD journey after disruption
    - new_distance - Float value of estimated distance of OD journey after disruption
    - new_path - List of string edge ID's of estimated new route of OD journey after disruption
    - new_time - Float value of estimated time of OD journey after disruption
    - dist_diff - Float value of Post disruption minus per-disruption distance
    - time_diff - Float value Post disruption minus per-disruption timee
    - min/max_tr_loss - Float value of estimated change in rerouting cost

- industry_columns - Float values of all daily tonnages of industry columns along disrupted OD pairs
- min/max_tons - Float values of total daily tonnages along disrupted OD pairs

2. Isolated OD scenarios - OD flows with no rerouting options
   - edge_id - String name or list of failed edges
   - o_region - String name of Province of Origin node ID of disrupted OD flow
   - d_region - String name of Province of Destination node ID of disrupted OD flow
   - industry_columns - Float values of all daily tonnages of industry columns along disrupted OD pairs
   - min/max_tons - Float values of total daily tonnages along disrupted OD pairs
3. Rerouting scenarios - OD flows with rerouting options
   - edge_id - String name or list of failed edges
   - o_region - String name of Province of Origin node ID of disrupted OD flow
   - d_region - String name of Province of Destination node ID of disrupted OD flow
   - min/max_tr_loss - Float value of change in rerouting cost
   - min/max_tons - Float values of total daily tonnages along disrupted OD pairs
4. Min-max combined scenarios - Combined min-max results along each edge
   - edge_id - String name or list of failed edges
   - no_access - Boolean 1 (no reroutng) or 0 (rerouting)
   - min/max_tr_loss - Float values of change in rerouting cost
   - min/max_tons - Float values of total daily tonnages affted by disrupted edge
5. **Shapefile Min-max combined scenarios - Combined min-max reults along each edge**
   - edge_id - String name or list of failed edges
   - no_access - Boolean 1 (no reroutng) or 0 (rerouting)
   - min/max_tr_loss - Float values of change in rerouting cost
   - min/max_tons - Float values of total daily tonnages affted by disrupted edge
   - geometry - Shapely LineString geomtry of edges

main()

Estimate failures

Specify the paths from where you want to read and write:

1. Input data
2. Intermediate calcuations data
3. Output results

Supply input data and parameters

1. **Names of modes** List of strings
2. **Unit weight of vehicle assumed for each mode** List of float types
3. **Range of usage factors for each mode to represent uncertainty in cost estimations** List of tuples of float types
4. **Min-max names of names of different types of attributes - paths, distance, time, cost, vehicles, tons** List of string types
5. **Names of commodity/industry columns for which min-max tonnage column names already exist** List of string types

6. **Percentage of OD flows that are assumed disrupted** List of float type
7. **Condition on whether analysis is single failure or multiple failure** Boolean condition True or False

Give the paths to the input data files:

1. Network edges Excel and shapefiles
2. OD flows Excel file
3. Costs of modes Excel file
4. Road properties Excel file
5. Failure scenarios Excel file

Specify the output files and paths to be created

atra.analysis.failure_estimation_dnv_flooded_roads module

Failure analysis of national-scale networks For transport modes at national scale:

- road
- rail

Input data requirements

1. Correct paths to all files and correct input parameters
2. csv sheets with results of flow mapping based on MIN-MAX generalised costs estimates:
    - origin - String node ID of Origin
    - destination - String node ID of Destination
    - origin_province - String name of Province of Origin node ID
    - destination_province - String name of Province of Destination node ID
    - min_edge_path - List of string of edge ID's for paths with minimum generalised cost flows
    - max_edge_path - List of string of edge ID's for paths with maximum generalised cost flows
    - min_distance - Float values of estimated distance for paths with minimum generalised cost flows
    - max_distance - Float values of estimated distance for paths with maximum generalised cost flows
    - min_time - Float values of estimated time for paths with minimum generalised cost flows
    - max_time - Float values of estimated time for paths with maximum generalised cost flows
    - min_gcost - Float values of estimated generalised cost for paths with minimum generalised cost flows
    - max_gcost - Float values of estimated generalised cost for paths with maximum generalised cost flows
    - industry_columns - All daily tonnages of industry columns given in the OD matrix data
3. Shapefiles
    - edge_id - String/Integer/Float Edge ID
    - geometry - Shapely LineString geomtry of edges

Results

Csv sheets with results of failure analysis:

1. All failure scenarios
    - edge_id - String name or list of failed edges

- origin - String node ID of Origin of disrupted OD flow
- destination - String node ID of Destination of disrupted OD flow
- origin_province - String name of Province of Origin node ID of disrupted OD flow
- destination_province - String name of Province of Destination node ID of disrupted OD flow
- no_access - Boolean 1 (no reroutng) or 0 (rerouting)
- min/max_distance - Float value of estimated distance of OD journey before disruption
- min/max_time - Float value of estimated time of OD journey before disruption
- min/max_gcost - Float value of estimated travel cost of OD journey before disruption
- new_cost - Float value of estimated cost of OD journey after disruption
- new_distance - Float value of estimated distance of OD journey after disruption
- new_path - List of string edge ID's of estimated new route of OD journey after disruption
- new_time - Float value of estimated time of OD journey after disruption
- dist_diff - Float value of Post disruption minus per-disruption distance
- time_diff - Float value Post disruption minus per-disruption timee
- min/max_tr_loss - Float value of estimated change in rerouting cost
- min/max_tons - Float values of total daily tonnages along disrupted OD pairs
- industry_columns - Float values of all daily tonnages of industry columns along disrupted OD pairs

2. Isolated OD scenarios - OD flows with no rerouting options
   - edge_id - String name or list of failed edges
   - origin_province - String name of Province of Origin node ID of disrupted OD flow
   - destination_province - String name of Province of Destination node ID of disrupted OD flow
   - industry_columns - Float values of all daily tonnages of industry columns along disrupted OD pairs
   - min/max_tons - Float values of total daily tonnages along disrupted OD pairs
3. Rerouting scenarios - OD flows with rerouting options
   - edge_id - String name or list of failed edges
   - origin_province - String name of Province of Origin node ID of disrupted OD flow
   - destination_province - String name of Province of Destination node ID of disrupted OD flow
   - min/max_tr_loss - Float value of change in rerouting cost
   - min/max_tons - Float values of total daily tonnages along disrupted OD pairs
4. Min-max combined scenarios - Combined min-max results along each edge
   - edge_id - String name or list of failed edges
   - no_access - Boolean 1 (no reroutng) or 0 (rerouting)
   - min/max_tr_loss - Float values of change in rerouting cost
   - min/max_tons - Float values of total daily tonnages affted by disrupted edge

main()

Estimate failures

Specify the paths from where you want to read and write:

1. Input data
2. Intermediate calcuations data

3. Output results

Supply input data and parameters

1. **Names of modes** String
2. **Names of min-max tons columns in sector data** List of string types
3. **Min-max names of names of different types of attributes - paths, distance, time, cost, tons** List of string types
4. **Names of commodity/industry columns for which min-max tonnage column names already exist** List of string types
5. **Percentage of OD flows that are assumed disrupted** List of float type
6. **Condition on whether analysis is single failure or multiple failure** Boolean condition True or False

Give the paths to the input data files:

1. Network edges csv and shapefiles
2. OD flows csv file
3. Failure scenarios csv file

Specify the output files and paths to be created

atra.analysis.flow_mapping module

Map flows on national networks

Purpose

Mapping the OD node level matrix values to network paths

For all transport modes at national scale: [‘road’, ‘rail’, ‘port’]

The code estimates 2 values - A MIN and a MAX value of flows between each selected OD node pair

- Based on MIN-MAX generalised costs estimates

Input data requirements

1. Correct paths to all files and correct input parameters
2. **Excel file with mode sheets containing network graph structure and attributes**
    - edge_id - String Edge ID
    - from_node - String node ID that should be present in node_id column
    - to_node - String node ID that should be present in node_id column
    - length - Float length of edge in km
    - min_time - Float minimum time of travel in hours on edge
    - max_time - Float maximum time of travel in hours on edge
    - min_time_cost - Float minimum cost of time in USD on edge
    - max_time_cost - Float maximum cost of time in USD on edge
    - min_tariff_cost - Float minimum tariff cost in USD on edge
    - max_tariff_cost - Float maximum tariff cost in USD on edge
3. **Edge shapefiles for all national-scale networks with attributes:**
    - edge_id - String Edge ID

- geometry - Shapely LineString geometry of edges

4. **Excel file with mode sheets containing node-level OD values with attributes:**
    - origin - String node ID of Origin
    - destination - String node ID of Destination
    - min_tons - Float values of minimum daily OD in tons
    - max_tons - Float values of maximum daily OD in tons
    - Names of the industry columns specified in the inputs

Results

1. **Excel sheets with results of flow mapping based on MIN-MAX generalised costs estimates:**
    - origin - String node ID of Origin
    - destination - String node ID of Destination
    - origin_province - String name of Province of Origin node ID
    - destination_province - String name of Province of Destination node ID
    - min_edge_path - List of string of edge ID's for paths with minimum generalised cost flows
    - max_edge_path - List of string of edge ID's for paths with maximum generalised cost flows
    - min_distance - Float values of estimated distance for paths with minimum generalised cost flows
    - max_distance - Float values of estimated distance for paths with maximum generalised cost flows
    - min_time - Float values of estimated time for paths with minimum generalised cost flows
    - max_time - Float values of estimated time for paths with maximum generalised cost flows
    - min_gcost - Float values of estimated generalised cost for paths with minimum generalised cost flows
    - max_gcost - Float values of estimated generalised cost for paths with maximum generalised cost flows
    - industry_columns - All daily tonnages of industry columns given in the OD matrix data
2. **Shapefiles**
    - edge_id - String/Integer/Float Edge ID
    - geometry - Shapely LineString geomtry of edges
    - min_{industry} - Float values of estimated minimum daily industries/commodities/total volumes in tons on edges
    - max_{industry} - Float values of estimated maximum daily industries/commodities/total volumes in tons on edges

References

1. Pant, R., Koks, E.E., Paltan, H., Russell, T. & Hall, J.W. (2019). Transport risk analysis in Argentina. Final Report, Oxford Infrastructure Analytics Ltd., Oxford, UK.
2. All input data folders and files referred to in the code below.

main()
Estimate flows

1. **Specify the paths from where you want to read and write:**
    - Input data
    - Intermediate calcuations data

- Output results

2. **Supply input data and parameters**
    - Percentage of OD flow we want to send along path: FLoat type
    - Names of modes: List of dictionaries
    - Names of min-max tonnage column names in OD data
3. **Give the paths to the input data files:**
    - Network edges csv files
    - OD daily flows csv file
4. Specify the output files and paths to be created

network_od_paths_assembly(*points_dataframe*, *graph*, *transport_mode*, *min_tons_column*, *max_tons_column*, *csv_output_path=''*)

Assemble estimates of OD paths, distances, times, costs and tonnages on networks

Parameters

- **points_dataframe** (*pandas.DataFrame*[36]) – OD nodes and their tonnages
- **graph** – igraph network structure
- **region_name** (*str*[37]) – name of Province
- **excel_writer** – Name of the excel writer to save Pandas dataframe to Excel file

Returns

save_paths_df –

- origin - String node ID of Origin
- destination - String node ID of Destination
- min_edge_path - List of string of edge ID's for paths with minimum generalised cost flows
- max_edge_path - List of string of edge ID's for paths with maximum generalised cost flows
- min_netrev - Float values of estimated netrevenue for paths with minimum generalised cost flows
- max_netrev - Float values of estimated netrevenue for paths with maximum generalised cost flows
- min_croptons - Float values of estimated crop tons for paths with minimum generalised cost flows
- max_croptons - Float values of estimated crop tons for paths with maximum generalised cost flows
- min_distance - Float values of estimated distance for paths with minimum generalised cost flows
- max_distance - Float values of estimated distance for paths with maximum generalised cost flows
- min_time - Float values of estimated time for paths with minimum generalised cost flows
- max_time - Float values of estimated time for paths with maximum generalised cost flows

[36] http://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.html#pandas.DataFrame
[37] https://docs.python.org/3.6/library/stdtypes.html#str

- min_gcost - Float values of estimated generalised cost for paths with minimum generalised cost flows
- max_gcost - Float values of estimated generalised cost for paths with maximum generalised cost flows

Return type pandas.DataFrame[38]

atra.analysis.hazards_network_intersections_results_collect module

Summarise network-hazard intersections

Purpose

Collect network-hazard intersection attributes

- Combine with boundary Polygons to collect network-hazard-boundary intersection attributes
- Write final results to an Excel sheet

Input data requirements

1. Correct paths to all files and correct input parameters
2. **Shapefiles of network-hazard intersections results with attributes:**
   - edge_id or node_id - String/Integer/Float Edge ID or Node ID of network
   - length - Float length of edge intersecting with hazards
   - geometry - Shapely geometry of edges as LineString or nodes as Points
3. **Shapefile of administrative boundaries of Argentina with attributes:**
   - province_i - String/Integer ID of Province
   - pro_name_e - String name of Province in English
   - district_i - String/Integer ID of District
   - dis_name_e - String name of District in English
   - commune_id - String/Integer ID of Commune
   - name_eng - String name of Commune in English
   - geometry - Shapely geometry of boundary Polygon
4. **Excel sheet of hazard attributes with attributes:**
   - hazard_type - String name of hazard type
   - model - String name of hazard model
   - year - String name of hazard year
   - climate_scenario - String name of hazard scenario
   - probability - Float/String value of hazard probability
   - band_num - Integer value of hazard band
   - min_val - Integer value of minimum value of hazard threshold
   - max_val - Integer value of maximum value of hazard threshold

[38] http://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.html#pandas.DataFrame

Results

1. **Excel sheet of network-hazard-boundary intersection with attributes:**
    - edge_id/node_id - String name of intersecting edge ID or node ID
    - length - Float length of intersection of edge LineString and hazard Polygon: Only for edges
    - province_id - String/Integer ID of Province
    - province_name - String name of Province in English
    - district_id - String/Integer ID of District
    - district_name - String name of District in English
    - commune_id - String/Integer ID of Commune
    - commune_name - String name of Commune in English
    - sector - String name of transport mode
    - hazard_type - String name of hazard type
    - model - String name of hazard model
    - year - String name of hazard year
    - climate_scenario - String name of hazard scenario
    - probability - Float/String value of hazard probability
    - band_num - Integer value of hazard band
    - min_val - Integer value of minimum value of hazard threshold
    - max_val - Integer value of maximum value of hazard threshold

create_hazard_attributes_for_network(*intersection_dir*, *climate_scenario*, *year*, *sector*, *hazard_files*, *hazard_df*, *thresholds*, *commune_shape*, *network_id_column*, *network_type=''*)

Extract results of network edges/nodes and hazard intersections to collect network-hazard intersection attributes

- Combine with boundary Polygons to collect network-hazard-boundary intersection attributes
- Write final results to an Excel sheet

Parameters

- **intersection_dir** (*str*[39]) – Path to Directory where the network-hazard shapefile results are stored
- **sector** (*str*[40]) – name of transport mode
- **hazard_files** (*list*[41] *[str*[42] *]*) – names of all hazard files
- **hazard_df** (*pandas.DataFrame*[43]) – hazard attributes
- **bands** (*list*[44] *[int*[45] *]*) – integer values of hazard bands
- **thresholds** (*list*[46] *[int*[47] *]*) – integer values of hazard thresholds

[39] https://docs.python.org/3.6/library/stdtypes.html#str
[40] https://docs.python.org/3.6/library/stdtypes.html#str
[41] https://docs.python.org/3.6/library/stdtypes.html#list
[42] https://docs.python.org/3.6/library/stdtypes.html#str
[43] http://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.html#pandas.DataFrame
[44] https://docs.python.org/3.6/library/stdtypes.html#list
[45] https://docs.python.org/3.6/library/functions.html#int
[46] https://docs.python.org/3.6/library/stdtypes.html#list
[47] https://docs.python.org/3.6/library/functions.html#int

- **commune_shape** – Shapefile of commune boundaries and attributes
- **network_type** (*str*[48], *optional*) – value -'edges' or 'nodes': Default = 'nodes'
- **name_province** (*str*[49], *optional*) – name of province if needed: Default = ''

Returns

data_df –

network-hazard-boundary intersection attributes:

- edge_id/node_id - String name of intersecting edge ID or node ID
- length - Float length of intersection of edge LineString and hazard Polygon: Only for edges
- province_id - String/Integer ID of Province
- province_name - String name of Province in English
- district_id - String/Integer ID of District
- district_name - String name of District in English
- commune_id - String/Integer ID of Commune
- commune_name - String name of Commune in English
- sector - String name of transport mode
- hazard_type - String name of hazard type
- model - String name of hazard model
- year - String name of hazard year
- climate_scenario - String name of hazard scenario
- probability - Float/String value of hazard probability
- band_num - Integer value of hazard band
- min_val - Integer value of minimum value of hazard threshold
- max_val - Integer value of maximum value of hazard threshold
- length - Float length of intersection of edge LineString and hazard Polygon: Only for edges

Return type pandas.DataFrame[50]

main()

Collect results

1. **Specify the paths from where you to read and write:**
   - Input data
   - Intermediate calcuations data
   - Output results
2. **Supply input data and parameters**
   - Names of the three Provinces - List of string types
   - Names of modes - List of strings
   - Names of output modes - List of strings

[48] https://docs.python.org/3.6/library/stdtypes.html#str
[49] https://docs.python.org/3.6/library/stdtypes.html#str
[50] http://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.html#pandas.DataFrame

- Names of hazard bands - List of integers
- Names of hazard thresholds - List of integers
- Condition 'Yes' or 'No' is the users wants to process results

3. **Give the paths to the input data files:**
    - Commune boundary and stats data shapefile
    - Hazard datasets description Excel file
    - String name of sheet in hazard datasets description Excel file

atra.analysis.hazards_networks_intersections module

Intersect networks with hazards

Purpose

Intersect hazards and network line and point geometries with hazatd polygons

Write final results to Shapefiles

Input data requirements

1. Correct paths to all files and correct input parameters
2. **Shapefiles of network edges or nodes with attributes:**
    - edge_id or node_id - String/Integer/Float Edge ID or Node ID of network
    - geometry - Shapely geometry of edges as LineStrings or nodes as Points
3. **Shapefile of hazards with attributes:**
    - geometry - Shapely geometry of hazard Polygon

Results

1. **Edge shapefiles with attributes:**
    - edge_id - String name of intersecting edge ID
    - length - Float length of intersection of edge LineString and hazard Polygon
    - geometry - Shapely LineString geometry of intersection of edge LineString and hazard Polygon
2. **Node Shapefile with attributes:**
    - node_id - String name of intersecting node ID
    - geometry - Shapely Point geometry of intersecting node ID

intersect_networks_and_all_hazards(*hazard_dir*, *network_file_path*, *network_file_name*, *output_file_path*, *network_id_column*, *network_type=''*)

Walk through all hazard files and select network-hazard intersection criteria

Parameters

- **hazard_dir** (*str*[51]) – name of directory where all hazard shapefiles are stored
- **network_file_path** (*str*[52]) – name of directory where network shapefile is stored
- **network_file_name** (*str*[53]) – name network shapefile

[51] https://docs.python.org/3.6/library/stdtypes.html#str
[52] https://docs.python.org/3.6/library/stdtypes.html#str
[53] https://docs.python.org/3.6/library/stdtypes.html#str

- **output_file_path** (*str*[54]) – name of directory where network-hazard instersection result shapefiles will be stored
- **network_type** (*str*[55]) – values of 'edges' or 'nodes'

Edge or Node shapefiles

main()

Intersect networks with hazards

1. **Specify the paths from where you to read and write:**
   - Input data
   - Intermediate calcuations data
   - Output results
2. **Supply input data and parameters**
   - Names of modes - List of strings
   - Names of mode id columns - List of strings
   - Condition 'Yes' or 'No' is the users wants to process results
3. **Give the paths to the input data files:**
   - Hazard directory
   - Paths to the network shapefiles

networkedge_hazard_intersection(*edge_shapefile*, *hazard_shapefile*, *output_shapefile*, *edge_id_column*)

Intersect network edges and hazards and write results to shapefiles

Parameters

- **edge_shapefile** – Shapefile of network LineStrings
- **hazard_shapefile** – Shapefile of hazard Polygons
- **output_shapefile** – String name of edge-hazard shapefile for storing results

output_shapefile

- edge_id - String name of intersecting edge ID
- length - Float length of intersection of edge LineString and hazard Polygon
- geometry - Shapely LineString geometry of intersection of edge LineString and hazard Polygon

networknode_hazard_intersection(*node_shapefile*, *hazard_shapefile*, *output_shapefile*, *node_id_column*)

Intersect network nodes and hazards and write results to shapefiles

Parameters

- **node_shapefile** – Shapefile of network Points
- **hazard_shapefile** – Shapefile of hazard Polygons
- **output_shapefile** – String name of node-hazard shapefile for storing results

output_shapefile

- node_id - String name of intersecting node ID
- geometry - Shapely Point geometry of intersecting node ID

[54] https://docs.python.org/3.6/library/stdtypes.html#str
[55] https://docs.python.org/3.6/library/stdtypes.html#str

atra.analysis.multi_modal_failure_estimation module

Failure analysis of national-scale networks For transport modes at national scale:

- rail
- Can do raod as well

Input data requirements

1. Correct paths to all files and correct input parameters
2. csv sheets with results of flow mapping based on MIN-MAX generalised costs estimates:
    - origin - String node ID of Origin
    - destination - String node ID of Destination
    - origin_province - String name of Province of Origin node ID
    - destination_province - String name of Province of Destination node ID
    - min_edge_path - List of string of edge ID's for paths with minimum generalised cost flows
    - max_edge_path - List of string of edge ID's for paths with maximum generalised cost flows
    - min_distance - Float values of estimated distance for paths with minimum generalised cost flows
    - max_distance - Float values of estimated distance for paths with maximum generalised cost flows
    - min_time - Float values of estimated time for paths with minimum generalised cost flows
    - max_time - Float values of estimated time for paths with maximum generalised cost flows
    - min_gcost - Float values of estimated generalised cost for paths with minimum generalised cost flows
    - max_gcost - Float values of estimated generalised cost for paths with maximum generalised cost flows
    - industry_columns - All daily tonnages of industry columns given in the OD matrix data
3. Shapefiles
    - edge_id - String/Integer/Float Edge ID
    - geometry - Shapely LineString geomtry of edges

Results

Csv sheets with results of failure analysis:

1. All failure scenarios
    - edge_id - String name or list of failed edges
    - origin - String node ID of Origin of disrupted OD flow
    - destination - String node ID of Destination of disrupted OD flow
    - origin_province - String name of Province of Origin node ID of disrupted OD flow
    - destination_province - String name of Province of Destination node ID of disrupted OD flow
    - no_access - Boolean 1 (no reroutng) or 0 (rerouting)
    - min/max_distance - Float value of estimated distance of OD journey before disruption
    - min/max_time - Float value of estimated time of OD journey before disruption
    - min/max_gcost - Float value of estimated travel cost of OD journey before disruption
    - new_cost - Float value of estimated cost of OD journey after disruption
    - new_distance - Float value of estimated distance of OD journey after disruption

- new_path - List of string edge ID's of estimated new route of OD journey after disruption
- new_time - Float value of estimated time of OD journey after disruption
- dist_diff - Float value of Post disruption minus per-disruption distance
- time_diff - Float value Post disruption minus per-disruption timee
- min/max_tr_loss - Float value of estimated change in rerouting cost
- min/max_tons - Float values of total daily tonnages along disrupted OD pairs
- industry_columns - Float values of all daily tonnages of industry columns along disrupted OD pairs

2. Isolated OD scenarios - OD flows with no rerouting options
   - edge_id - String name or list of failed edges
   - origin_province - String name of Province of Origin node ID of disrupted OD flow
   - destination_province - String name of Province of Destination node ID of disrupted OD flow
   - industry_columns - Float values of all daily tonnages of industry columns along disrupted OD pairs
   - min/max_tons - Float values of total daily tonnages along disrupted OD pairs
3. Rerouting scenarios - OD flows with rerouting options
   - edge_id - String name or list of failed edges
   - origin_province - String name of Province of Origin node ID of disrupted OD flow
   - destination_province - String name of Province of Destination node ID of disrupted OD flow
   - min/max_tr_loss - Float value of change in rerouting cost
   - min/max_tons - Float values of total daily tonnages along disrupted OD pairs
4. Min-max combined scenarios - Combined min-max results along each edge
   - edge_id - String name or list of failed edges
   - no_access - Boolean 1 (no reroutng) or 0 (rerouting)
   - min/max_tr_loss - Float values of change in rerouting cost
   - min/max_tons - Float values of total daily tonnages affted by disrupted edge

main()

Estimate failures

Specify the paths from where you want to read and write:

1. Input data
2. Intermediate calcuations data
3. Output results

Supply input data and parameters

1. **Names of modes** String
2. **Names of min-max tons columns in sector data** List of string types
3. **Min-max names of names of different types of attributes - paths, distance, time, cost, tons** List of string types
4. **Names of commodity/industry columns for which min-max tonnage column names already exist** List of string types
5. **Percentage of OD flows that are assumed disrupted** List of float type
6. **Condition on whether analysis is single failure or multiple failure** Boolean condition True or False

Give the paths to the input data files:

1. Network edges csv and shapefiles
2. OD flows csv file
3. Failure scenarios csv file

Specify the output files and paths to be created

atra.analysis.risk_calculations module

Road network risks and adaptation maps

main()

atra.mrio package

Submodules

atra.mrio.create_mrio module

atra.mrio.ras_method module

RAS Method

Purpose

Estimate a new matrix X with exogenously given row and column totals that is a close as possible to a given original matrix X0 using the Generalized RAS (GRAS) approach

Usage

X = gras(X0, u, v) OR [X, r, s] = gras(X0, u, v) with or without eps included as the fourth argument, where

Input

- X0 = benchmark (base) matrix, not necessarily square
- u = column vector of (new) row totals
- v = column vector of (new) column totals
- eps = convergence tolerance level; if empty, the default threshold is 0.1e-5 (=0.000001)

Output

- X = estimated/adjusted/updated matrix
- r = substitution effects (row multipliers)
- s = fabrication effects (column multipliers)

References

1) Junius T. and J. Oosterhaven (2003), The solution of updating or regionalizing a matrix with both positive and negative entries, Economic Systems Research, 15, pp. 87-96.

2) Lenzen M., R. Wood and B. Gallego (2007), Some comments on the GRAS method, Economic Systems Research, 19, pp. 461-465.

3) Temurshoev, U., R.E. Miller and M.C. Bouwmeester (2013), A note on the GRAS method, Economic Systems Research, 25, pp. 361-367.

invd(*x*)

ras_method(*X0*, *u*, *v*, *eps=1e-05*, *print_out=False*)

atra.plot package

Submodules

atra.plot.adaptation_sensitvity module

Plot adaptation cost ranges (national results)

adaptation_sentivity_plot(*input_dataframe*, *x_col*, *y_col*, *z_col*, *x_label*, *y_label*, *plot_title*, *plot_path*, *mode*)

main()

atra.plot.admin_map module

Plot country and administrative areas

main(*config*)
Read shapes, plot map

atra.plot.air_network_flows module

air network flows map

main()

atra.plot.ba_hazard_maps module

Plot country and administrative areas

main(*config*)
Read shapes, plot map

atra.plot.bridge_failures module

Road network failure maps

main()

atra.plot.bridge_risks module

Road network risks and adaptation maps

main()

atra.plot.bridge_risks_combined module

Road network risks and adaptation maps

main()

atra.plot.bridges module

Plot road network

main(*config*)
Read shapes, plot map

atra.plot.bridges_adaptation module

Road network risks and adaptation maps

main()

atra.plot.change_plots module

atra.plot.climate_change_plots module

Plot adaptation cost ranges (national results)

main()

plot_values(*input_data*, *index_column*, *index_values*, *x_column*, *y_column*, *division_factor*, *x_label*, *y_label*, *plot_title*, *plot_colors*, *plot_markers*, *plot_file_path*)

atra.plot.cost_vs_projects module

atra.plot.hazard_maps module

Plot country and administrative areas

main(*config*)
Read shapes, plot map

atra.plot.national_hazard_exposure_plots module

National hazard exposure maps

main()

atra.plot.national_rail_risks module

rail network risks and adaptation maps

main()

atra.plot.national_rail_risks_combined module

rail network risks and adaptation maps

main()

atra.plot.national_roads_risks module

Road network risks and adaptation maps

main()

atra.plot.national_roads_risks_combined module

Road network risks and adaptation maps

main()

atra.plot.network_air module

Plot air network

main(*config*)
Read shapes, plot map

atra.plot.network_rail module

Plot rail network

main(*config*)
Read shapes, plot map

atra.plot.network_road module

Plot road network

main(*config*)
Read shapes, plot map

atra.plot.network_road_rural module

Plot road network

main(*config*)
Read shapes, plot map

atra.plot.network_water module

Plot water network

main(*config*)
Read shapes, plot map

atra.plot.od_commodities_charts module

Plot commodities matrices

main(*config*)
Read data, plot charts

atra.plot.od_commodities_maps module

atra.plot.plot_ranges module

Plot adaptation cost ranges (national results)

main()

plot_many_ranges(*input_dfs*, *division_factor*, *x_label*, *y_label*, *plot_title*, *plot_color*, *plot_labels*, *plot_file_path*)

plot_many_ranges_subplots(*input_dfs*, *division_factor*, *x_label*, *y_label*, *plot_title*, *plot_color*, *plot_labels*, *plot_file_path*)

plot_ranges(*input_data*, *division_factor*, *x_label*, *y_label*, *plot_title*, *plot_color*, *plot_file_path*)

atra.plot.population_maps module

Plot country and administrative areas

main(*config*)
Read shapes, plot map

atra.plot.rail_failures_multi_modal module

Network rerouting loss maps

change_to_infinity(*x*, *dividend_column*, *divisor_column*)

main()

plot_many_ranges(*input_dfs*, *division_factor*, *x_label*, *y_label*, *plot_title*, *plot_color*, *plot_labels*, *plot_file_path*)

plot_ranges(*input_data*, *division_factor*, *x_label*, *y_label*, *plot_title*, *plot_color*, *plot_file_path*, *ylimit=None*, *yticks_loc=None*, *y_ticks_labels=None*)

atra.plot.rail_network_failures module

Rail network loss maps

main()

atra.plot.rail_network_flows_max_scales module

Road network flow maps

main()

atra.plot.rail_routes module

Rail network routes map

main()

atra.plot.risk_changes module

atra.plot.risk_comparison module

Road network risks and adaptation maps

main()

plot_many_ranges(*input_dfs*, *division_factor*, *x_label*, *y_label*, *plot_title*, *plot_color*, *plot_labels*, *plot_file_path*)

atra.plot.road_dnv_estimates module

Road network flow maps

assign_veh_to_roads(*x*, *veh_list*)

Assign terrain as flat or mountain to national roads

Parameters

x - Pandas DataFrame of values

- **dia_hinh__** - String value of type of terrain

Returns String value of terrain as flat or mountain

main()

atra.plot.road_network_failures module

Road network failure maps

main()

atra.plot.road_network_flows_max_scales module

Road network flow maps

main()

atra.plot.road_tmda_flows_correlations module

atra.plot.roads_adaptation module

Road network risks and adaptation maps

main()

atra.plot.water_network_flows_max_scales module

Coastal network flows map

main()

atra.preprocess package

Submodules

atra.preprocess.combine_roads module

Get Argentina shapefiles and combine them into a single file

main()

atra.preprocess.convert_hazard_data module

Pre-process hazard data

Purpose

Convert GeoTiff raster hazard datasets to shapefiles based on masking and selecting values from

- Single-band raster files

Input data requirements

1. Correct paths to all hazard datasets
2. **Single-band GeoTiff hazard raster files with:**
    - values - between 0 and 1000
    - raster grid geometry
    - projection systems: Default assumed = EPSG:4326

Results

1. **Shapefiles whose names show the hazard models and their selected range of values**
    - ID - equal to 1
    - geometry - Shapely Polygon outline of selected hazard

convert(*threshold*, *infile*, *tmpfile_1*, *outfile*)

Convert GeoTiff raster file to Shapefile with geometries based on raster threshold less that 999

Parameters

- threshold - Float value of lower bound of GeoTiff threshold value to be selected
- infile - String name of input GeoTff file path
- tmpfile_1 - Stirng name of tmp file 1
- outfile - Stirng name of output shapefile

Outputs Shapefile with Polygon geometries of rasters based on raster values above a threshold

convert_geotiff_to_vector_with_multibands(*band_colors*, *infile*, *infile_epsg*, *tmpfile_1*, *tmpfile_2*, *outfile*)

Convert multi-band GeoTiff raster file to Shapefile with geometries based on raster band color values

Parameters

- band_colors - Tuple with 3-values each corresponding to the values in raster bands
- infile - String name of input GeoTff file path
- infile_epsg - Integer value of EPSG Projection number of raster
- tmpfile_1 - Stirng name of tmp file 1
- tmpfile_2 - Stirng name of tmp file 2
- outfile - Stirng name of output shapefile

Outputs Shapefile with Polygon geometries of rasters based on raster band values

convert_geotiff_to_vector_with_threshold(*from_threshold*, *to_threshold*, *infile*, *infile_epsg*, *tmpfile_1*, *tmpfile_2*, *outfile*)

Convert GeoTiff raster file to Shapefile with geometries based on raster threshold ranges

Parameters

- from_threshold - Float value of lower bound of GeoTiff threshold value to be selected
- to_threshold - Float value of upper bound of GeoTiff threshold value to be selected
- infile - String name of input GeoTff file path
- infile_epsg - Integer value of EPSG Projection number of raster
- tmpfile_1 - Stirng name of tmp file 1
- tmpfile_2 - Stirng name of tmp file 2
- outfile - Stirng name of output shapefile

Outputs Shapefile with Polygon geometries of rasters based on raster threshold ranges

glofris_data_details(*file_name*, *root_dir*)

Read names of GLOFRIS files and create attributes

Parameters

- file_name - String name of GeoTff file
- root_dir - String path to directory of file

Outputs

df - Pandas DataFrame written to csv file with columns:

- file_name - String
- hazard_type - String
- year - Integer: 2016 or 2030
- climate_scenario - String: RCP4.5 or RCP8.5 or none
- probability - Float: 1/(return period)
- banded - Boolean: True or False
- bands - Integer

main()

Process hazard data

1. **Specify the paths from where to read and write:**
   - Input data

- Hazard data

2. **Supply input data and parameters**
    - Thresholds of flood hazards
    - Values of bands to be selected
    - Color code of multi-band rasters
    - Specific file names that might require some specific operations

raster_projections_and_databands(*file_path*)
Extract projection, data bands numbers and valuees from raster

Parameters

- file_path - String name of input GeoTff file path

Outputs

- counts - Number of bans in raster
- crs - Projection system of raster
- data_vals - Numpy array of raster values

raster_rewrite(*in_raster*, *out_raster*, *nodata*)
Rewrite a raster to reproject and change no data value

Parameters

- in_raster - String name of input GeoTff file path
- out_raster - String name of output GeoTff file path
- nodata - Float value of data that is treated as no data

Outputs Reproject and replace raster with nodata = -1

atra.preprocess.multi_modal_network_creation module

atra.preprocess.network_air module

Create air network and passenger usage data for Argentina

main(*config*)

atra.preprocess.network_road_topology module

atra.preprocess.od_combine module

atra.preprocess.port_od_flows module

atra.preprocess.rail_od_flows module

atra.preprocess.road_bridge_matches module

atra.preprocess.road_network_creation module

atra.preprocess.road_od_flows module

atra.preprocess.scrape_wfs module

atra.stats package

Submodules

atra.stats.air_water_vulnerability_stats module

Sum max/min total flow exposed under hazard scenarios at air and water network nodes

join_hazards(*nodes_with_flows_df*, *hazards_df*)

main()

read_hazards(*hazard_file*)

summarise(*nodes_with_hazards_df*, *id_columns*, *sort_column*, *mode*)

atra.stats.boundary_hazard_percentages module

Summarise network-hazard intersections per-boundary (district, commune or province)

Purpose

Collect network-hazard intersection attributes

- Combine with boundary Polygons to collect network-boundary intersection attributes
- Write final results to an Excel sheet

Input data requirements

1. Correct paths to all files and correct input parameters
2. **Shapefiles of network-hazard intersections results with attributes:**
    - edge_id or node_id - String/Integer/Float Edge ID or Node ID of network
    - length - Float length of edge intersecting with hazards
    - geometry - Shapely geometry of edges as LineString or nodes as Points
3. **Shapefile of administrative boundaries of Argentina with attributes:**
    - province_i - String/Integer ID of Province
    - pro_name_e - String name of Province in English
    - district_i - String/Integer ID of District
    - dis_name_e - String name of District in English
    - commune_id - String/Integer ID of Commune
    - name_eng - String name of Commune in English
    - geometry - Shapely geometry of boundary Polygon

Results

1. **Excel sheet of network-hazard-boundary intersection with attributes:**
    - edge_id/node_id - String name of intersecting edge ID or node ID
    - length - Float length of intersection of edge LineString and hazard Polygon: Only for edges
    - province_id - String/Integer ID of Province
    - province_name - String name of Province in English
    - district_id - String/Integer ID of District
    - district_name - String name of District in English
    - commune_id - String/Integer ID of Commune
    - commune_name - String name of Commune in English

main()

Summarise intersections

1. **Specify the paths from where you to read and write:**
    - Input data
    - Intermediate calcuations data
    - Output results
2. **Supply input data and parameters**
    - Names of the three Provinces - List of string types
    - Names of modes - List of strings
    - Names of output modes - List of strings
    - Names of hazard bands - List of integers
    - Names of hazard thresholds - List of integers
    - Condition 'Yes' or 'No' is the users wants to process results
3. **Give the paths to the input data files:**
    - Commune boundary and stats data shapefile
    - Hazard datasets description Excel file
    - String name of sheet in hazard datasets description Excel file
4. Specify the output files and paths to be created

atra.stats.flow_senstivities module

Summarise hazard data

Get OD data and process it

main()

nodes_flows_from_edges(*edge_flow_file*, *nodes_name_file*, *nodes_name_column*, *flow_columns*)

atra.stats.network_boundary_stats module

atra.stats.network_failure_stats module

Summarise length of edges/number of nodes within each boundary (commune, district, province)

Purpose

Collect network attributes

- Combine with boundary Polygons to collect network-boundary intersection attributes
- Write final results to an Excel sheet

Input data requirements

1. Correct paths to all files and correct input parameters
2. **Shapefiles of networks with attributes:**
    - edge_id or node_id - String/Integer/Float Edge ID or Node ID of network
    - length - Float length of edge intersecting with hazards
    - geometry - Shapely geometry of edges as LineString or nodes as Points

3. **Shapefile of administrative boundaries of Argentina with attributes:**
    - province_i - String/Integer ID of Province
    - pro_name_e - String name of Province in English
    - district_i - String/Integer ID of District
    - dis_name_e - String name of District in English
    - commune_id - String/Integer ID of Commune
    - name_eng - String name of Commune in English
    - geometry - Shapely geometry of boundary Polygon

Results

1. **Excel sheet of network-hazard-boundary intersection with attributes:**
    - edge_id/node_id - String name of intersecting edge ID or node ID
    - length - Float length of intersection of edge LineString and hazard Polygon: Only for edges
    - province_id - String/Integer ID of Province
    - province_name - String name of Province in English
    - district_id - String/Integer ID of District
    - district_name - String name of District in English
    - commune_id - String/Integer ID of Commune
    - commune_name - String name of Commune in English

change_matrix(*risk_dataframe*, *value_threshold*, *change_threshold*)

main()

Summarise

1. **Specify the paths from where you to read and write:**
    - Input data
    - Intermediate calcuations data
    - Output results
2. **Supply input data and parameters**
    - Names of the three Provinces - List of string types
    - Names of modes - List of strings
    - Names of output modes - List of strings
    - Names of hazard bands - List of integers
    - Names of hazard thresholds - List of integers
    - Condition 'Yes' or 'No' is the users wants to process results
3. **Give the paths to the input data files:**
    - Commune boundary and stats data shapefile
    - String name of sheet in hazard datasets description Excel file
4. Specify the output files and paths to be created

risk_results_reorganise(*risk_dataframe*, *id_column*)

risk_results_reorganise_climate_outlooks(*risk_dataframe*, *id_column*)

atra.stats.network_hazard_stats module

Summarise per-hazard total intersections (for the whole system)

Purpose

Collect network-hazard intersection attributes

- Combine with boundary Polygons to collect network-boundary intersection attributes
- Write final results to an Excel sheet

Input data requirements

1. Correct paths to all files and correct input parameters
2. **Shapefiles of network-hazard intersections results with attributes:**
    - edge_id or node_id - String/Integer/Float Edge ID or Node ID of network
    - length - Float length of edge intersecting with hazards
    - geometry - Shapely geometry of edges as LineString or nodes as Points
3. **Shapefile of administrative boundaries of Argentina with attributes:**
    - province_i - String/Integer ID of Province
    - pro_name_e - String name of Province in English
    - district_i - String/Integer ID of District
    - dis_name_e - String name of District in English
    - commune_id - String/Integer ID of Commune
    - name_eng - String name of Commune in English
    - geometry - Shapely geometry of boundary Polygon

Results

1. **Excel sheet of network-hazard-boundary intersection with attributes:**
    - edge_id/node_id - String name of intersecting edge ID or node ID
    - length - Float length of intersection of edge LineString and hazard Polygon: Only for edges
    - province_id - String/Integer ID of Province
    - province_name - String name of Province in English
    - district_id - String/Integer ID of District
    - district_name - String name of District in English
    - commune_id - String/Integer ID of Commune
    - commune_name - String name of Commune in English

hazard_data_summary(*hazard_network_dataframe*, *network_dataframe*)

main()
Summarise

1. **Specify the paths from where you to read and write:**
    - Input data
    - Intermediate calcuations data
    - Output results

2. **Supply input data and parameters**
    - Names of the three Provinces - List of string types
    - Names of modes - List of strings
    - Names of output modes - List of strings
    - Names of hazard bands - List of integers
    - Names of hazard thresholds - List of integers
    - Condition 'Yes' or 'No' is the users wants to process results
3. **Give the paths to the input data files:**
    - Commune boundary and stats data shapefile
    - Hazard datasets description Excel file
    - String name of sheet in hazard datasets description Excel file
4. Specify the output files and paths to be created

atra.stats.network_stats module

Summarise length of edges/number of nodes within each boundary (commune, district, province)

Purpose

Collect network attributes

- Combine with boundary Polygons to collect network-boundary intersection attributes
- Write final results to an Excel sheet

Input data requirements

1. Correct paths to all files and correct input parameters
2. **Shapefiles of networks with attributes:**
    - edge_id or node_id - String/Integer/Float Edge ID or Node ID of network
    - length - Float length of edge intersecting with hazards
    - geometry - Shapely geometry of edges as LineString or nodes as Points
3. **Shapefile of administrative boundaries of Argentina with attributes:**
    - province_i - String/Integer ID of Province
    - pro_name_e - String name of Province in English
    - district_i - String/Integer ID of District
    - dis_name_e - String name of District in English
    - commune_id - String/Integer ID of Commune
    - name_eng - String name of Commune in English
    - geometry - Shapely geometry of boundary Polygon

Results

1. **Excel sheet of network-hazard-boundary intersection with attributes:**
    - edge_id/node_id - String name of intersecting edge ID or node ID
    - length - Float length of intersection of edge LineString and hazard Polygon: Only for edges

- province_id - String/Integer ID of Province
- province_name - String name of Province in English
- district_id - String/Integer ID of District
- district_name - String name of District in English
- commune_id - String/Integer ID of Commune
- commune_name - String name of Commune in English

main()

Summarise

1. **Specify the paths from where you to read and write:**
   - Input data
   - Intermediate calcuations data
   - Output results
2. **Supply input data and parameters**
   - Names of the three Provinces - List of string types
   - Names of modes - List of strings
   - Names of output modes - List of strings
   - Names of hazard bands - List of integers
   - Names of hazard thresholds - List of integers
   - Condition 'Yes' or 'No' is the users wants to process results
3. **Give the paths to the input data files:**
   - Commune boundary and stats data shapefile
   - String name of sheet in hazard datasets description Excel file
4. Specify the output files and paths to be created

1.6.2 Submodules

1.6.3 atra.adaptation_options module

Estimate costs and benefits under fixed parameters varying the

cost components, durations of disruptions, GDP growth rates

calc_benefits_and_bcr(*x*, *discount_rates*, *discount_growth_rates*, *duration_max=10*, *min_loss=True*, *mode='road'*)

Estimate the total cost and benefits for a road segment. This function is used within a pandas apply

Parameters

- **x** – a row from the road segment dataframe that we are considering
- **param_values** – numpy array with a set of parameter combinations
- **mnt_dis_cost** – adaptation costs for a district road in the mountains
- **mnt_nat_cost** – adaptation costs for a national road in the mountains
- **cst_dis_cost** – adaptation costs for a district road on flat terrain
- **cst_nat_cost** – adaptation costs for a national road on flat terrain
- **pavement** – set of paving combinations. This corresponds with the cost table and the param_values
- **mnt_main_cost** – maintenance costs for roads in the mountains

- **cst_main_cost** – maintenance costs for roads on flat terrain
- **discount_rates** – discount rates to be used for the costs
- **discount_growth_rates** – discount rates to be used for the losses
- **rehab_costs** – rehabilitation costs after a disaster
- **min_main_dr** – discount rates for 4-year periodic maintenance
- **max_main_dr** – discount rates for 8-year periodic maintenance
- **min_exp** (*bool*[56], *optional*) – Specify whether we want to use the minimum or maximum exposure length. The default value is set to **True**
- **national** (*bool*[57], *optional*) – Specify whether we are looking at national roads. The default value is set to **False**
- **min_loss** (*bool*[58], *optional*) – Specify whether we want to use the minimum or maximum economic losses. The default value is set to **True**

Returns

- **uncer_output** (*list*) – outcomes for the initial adaptation costs of this road segment
- **tot_uncer_output** (*list*) – outcomes for the total adaptation costs of this road segment
- **rel_share** (*list*) – relative share of each factor in the initial adaptation cost of this road segment
- **tot_rel_share** (*list*) – relative share of each factor in the total adaptation cost of this road segment
- **bc_ratio** (*list*) – benefit cost ratios for this road segment

calc_costs(*x*, *cst_2L_asphalt*, *cst_2L_concrete*, *cst_4L_concrete*, *cst_rehab*, *cst_routine*, *cst_periodic*, *discount_rates*, *min_main_dr*, *max_main_dr*, *mode='road'*)

Estimate the total cost and benefits for a road segment. This function is used within a pandas apply

Parameters

- **x** – a row from the road segment dataframe that we are considering
- **param_values** – numpy array with a set of parameter combinations
- **mnt_dis_cost** – adaptation costs for a district road in the mountains
- **mnt_nat_cost** – adaptation costs for a national road in the mountains
- **cst_dis_cost** – adaptation costs for a district road on flat terrain
- **cst_nat_cost** – adaptation costs for a national road on flat terrain
- **pavement** – set of paving combinations. This corresponds with the cost table and the param_values
- **mnt_main_cost** – maintenance costs for roads in the mountains
- **cst_main_cost** – maintenance costs for roads on flat terrain
- **discount_rates** – discount rates to be used for the costs
- **discount_growth_rates** – discount rates to be used for the losses
- **rehab_costs** – rehabilitation costs after a disaster
- **min_main_dr** – discount rates for 4-year periodic maintenance
- **max_main_dr** – discount rates for 8-year periodic maintenance

[56] https://docs.python.org/3.6/library/functions.html#bool
[57] https://docs.python.org/3.6/library/functions.html#bool
[58] https://docs.python.org/3.6/library/functions.html#bool

- **min_exp** (*bool*[59], *optional*) – Specify whether we want to use the minimum or maximum exposure length. The default value is set to **True**
- **national** (*bool*[60], *optional*) – Specify whether we are looking at national roads. The default value is set to **False**
- **min_loss** (*bool*[61], *optional*) – Specify whether we want to use the minimum or maximum economic losses. The default value is set to **True**

Returns

- **uncer_output** (*list*) – outcomes for the initial adaptation costs of this road segment
- **tot_uncer_output** (*list*) – outcomes for the total adaptation costs of this road segment
- **rel_share** (*list*) – relative share of each factor in the initial adaptation cost of this road segment
- **tot_rel_share** (*list*) – relative share of each factor in the total adaptation cost of this road segment
- **bc_ratio** (*list*) – benefit cost ratios for this road segment

calculate_discounting_arrays(*discount_rate=12*, *growth_rate=2.7*, *start_year=2016*, *end_year=2050*, *min_period=4*, *max_period=8*)

Set discount rates for yearly and period maintenance costs

Parameters

- **discount_rate** – yearly discount rate
- **growth_rate** – yearly growth rate

Returns

- *discount_rate_norm* – discount rates to be used for the costs
- *discount_rate_growth* – discount rates to be used for the losses
- *min_main_dr* – discount rates for 4-year periodic maintenance
- *max_main_dr* – discount rates for 8-year periodic maintenance

get_adaptation_options_costs(*file_id*, *data_path*, *output_path*, *results_type*, *discount_rate=10*, *start_year=2016*, *end_year=2050*, *min_period=4*, *max_period=8*, *read_from_file=False*)

run_adaptation_calculation(*roads*, *file_id*, *output_path*, *file_id_col*, *results_type_index_col*, *results_type*, *duration_max=10*, *discount_rate=10*, *growth_rate=2.8*, *start_year=2016*, *end_year=2050*, *min_period=4*, *max_period=8*, *read_from_file=False*)

1.6.4 atra.network module

Network representation and utilities

class Network(*nodes=None*, *edges=None*)

Bases: object[62]

A Network is composed of nodes (points in space) and edges (lines)

Parameters

- **nodes** (*geopandas.geodataframe.GeoDataFrame*, *optional*) –
- **edges** (*geopandas.geodataframe.GeoDataFrame*, *optional*) –

[59] https://docs.python.org/3.6/library/functions.html#bool
[60] https://docs.python.org/3.6/library/functions.html#bool
[61] https://docs.python.org/3.6/library/functions.html#bool
[62] https://docs.python.org/3.6/library/functions.html#object

nodes

Type geopandas.geodataframe.GeoDataFrame

edges

Type geopandas.geodataframe.GeoDataFrame

set_crs(*crs=None*, *epsg=None*)
Set network (node and edge) crs

Parameters

- **crs** (*dict*[63] *or str*[64]) – Projection parameters as PROJ4 string or in dictionary form.
- **epsg** (*int*[65]) – EPSG code specifying output projection

to_crs(*crs=None*, *epsg=None*)
Set network (node and edge) crs

Parameters

- **crs** (*dict*[66] *or str*[67]) – Projection parameters as PROJ4 string or in dictionary form.
- **epsg** (*int*[68]) – EPSG code specifying output projection

add_endpoints(*network*)
Add nodes at line endpoints

add_ids(*network*, *id_col='id'*, *edge_prefix='edge'*, *node_prefix='node'*, *update=False*)
Add an id column with ascending ids

add_topology(*network*, *id_col='id'*, *update=False*)
Add from_id, to_id to edges

add_vertex(*line*, *point*)
Add a vertex to a line at a point

concat_dedup(*dfs*)
Concatenate a list of GeoDataFrames, dropping duplicate geometries - note: repeatedly drops indexes for deduplication to work

d_within(*geom*, *gdf*, *distance*)
Find the subset of a GeoDataFrame within some distance of a shapely geometry

drop_duplicate_geometries(*gdf*, *keep='first'*)
Drop duplicate geometries from a dataframe

edges_within(*point*, *edges*, *distance*)
Find edges within a distance of point

geometry_column_name(*gdf*)
Get geometry column name, fall back to 'geometry'

get_endpoints(*network*)
Get nodes for each edge endpoint

intersects(*geom*, *gdf*, *tolerance=1e-09*)
Find the subset of a GeoDataFrame intersecting with a shapely geometry

line_endpoints(*line*)
Return points at first and last vertex of a line

[63] https://docs.python.org/3.6/library/stdtypes.html#dict
[64] https://docs.python.org/3.6/library/stdtypes.html#str
[65] https://docs.python.org/3.6/library/functions.html#int
[66] https://docs.python.org/3.6/library/stdtypes.html#dict
[67] https://docs.python.org/3.6/library/stdtypes.html#str
[68] https://docs.python.org/3.6/library/functions.html#int

link_nodes_to_edges_within (*network*, *distance*, *condition=None*, *tolerance=1e-09*)
Link nodes to all edges within some distance

link_nodes_to_nearest_edge (*network*, *condition=None*)
Link nodes to all edges within some distance

matching_gdf_from_geoms (*gdf*, *geoms*)
Create a geometry-only GeoDataFrame with column name to match an existing GeoDataFrame

merge_edges (*network*)
Merge edges that share a node with a connectivity degree of 2

merge_multilinestring (*geom*)
Merge a MultiLineString to LineString

nearest (*geom*, *gdf*)
Find the element of a GeoDataFrame nearest a shapely geometry

nearest_edge (*point*, *edges*)
Find nearest edge to a point

nearest_node (*point*, *nodes*)
Find nearest node to a point

nearest_point_on_edges (*point*, *edges*)
Find nearest point on edges to a point

nearest_point_on_line (*point*, *line*)
Return the nearest point on a line

nearest_vertex_idx_on_line (*point*, *line*)
Return the index of nearest vertex to a point on a line

node_connectivity_degree (*node*, *network*)

nodes_intersecting (*line*, *nodes*, *tolerance=1e-09*)
Find nodes intersecting line

round_geometries (*network*, *precision=3*)
Round coordinates of all node points and vertices of edge linestrings to some precision

set_precision (*geom*, *precision*)
Set geometry precision

snap_line (*line*, *points*, *tolerance=1e-09*)
Snap a line to points within tolerance, inserting vertices as necessary

snap_nodes (*network*, *threshold=None*)
Move nodes (within threshold) to edges

split_edge_at_points (*edge*, *points*, *tolerance=1e-09*)
Split edge at point/multipoint

split_edges_at_nodes (*network*, *tolerance=1e-09*)
Split network edges where they intersect node geometries

split_line (*line*, *points*, *tolerance=1e-09*)
Split line at point or multipoint, within some tolerance

split_multilinestrings (*network*)
Create multiple edges from any MultiLineString edge

Ensures that edge geometries are all LineStrings, duplicates attributes over any created multi-edges.

1.6.5 atra.transport_flow_and_failure_functions module

Functions used in the provincial and national-scale network failure analysis

add_dataframe_generalised_costs (*G*, *vehicle_numbers*, *tonnage*)

add_igraph_generalised_costs(*G*, *vehicle_numbers*, *tonnage*)

change_depth_string_to_number(*x*)

combine_hazards_and_network_attributes_and_impacts(*hazard_dataframe*, *network_dataframe*, *network_id_column*)

correct_exposures(*x*, *length_thr*)

create_hazard_scenarios_for_adaptation(*all_edge_fail_scenarios*, *index_cols*, *length_thr*)

edge_failure_sampling(*failure_scenarios*, *edge_column*)
Criteria for selecting failure samples

Parameters

- **– Pandas DataFrame of failure scenarios** (*failure_scenarios*) –
- **– String name of column to select failed edge ID's** (*edge_column*) –

Returns

Return type edge_failure_samples - List of lists of failed edge sets

get_flow_paths_indexes_of_edges(*flow_dataframe*, *path_criteria*)

igraph_scenario_edge_failures_new(*network_df_in*, *edge_failure_set*, *flow_dataframe*, *edge_flow_path_indexes*, *path_criteria*, *tons_criteria*, *cost_criteria*, *time_criteria*, *transport_mode*, *new_path=True*)
Estimate network impacts of each failures When the tariff costs of each path are fixed by vehicle weight

Parameters

- **– Pandas DataFrame of network** (*network_df_in*) –
- **– List of string edge ID's** (*edge_failure_set*) –
- **– Pandas DataFrame of list of edge paths** (*flow_dataframe*) –
- **– String name of column of edge paths in flow dataframe** (*path_criteria*) –
- **– String name of column of path tons in flow dataframe** (*tons_criteria*) –
- **– String name of column of path costs in flow dataframe** (*cost_criteria*) –
- **– String name of column of path travel time in flow dataframe** (*time_criteria*) –

Returns **edge_failure_dictionary** – With attributes edge_id - String name or list of failed edges origin - String node ID of Origin of disrupted OD flow destination - String node ID of Destination of disrupted OD flow no_access - Boolean 1 (no reroutng) or 0 (rerouting) new_cost - Float value of estimated cost of OD journey after disruption new_distance - Float value of estimated distance of OD journey after disruption new_path - List of string edge ID's of estimated new route of OD journey after disruption new_time - Float value of estimated time of OD journey after disruption

Return type list[69][dict[70]]

merge_failure_results(*flow_df_select*, *failure_df*, *id_col*, *tons_col*, *dist_col*, *time_col*, *cost_col*)
Merge failure results with flow results

[69] https://docs.python.org/3.6/library/stdtypes.html#list
[70] https://docs.python.org/3.6/library/stdtypes.html#dict

Parameters

- **flow_df_select** (*pandas.DataFrame*[71]) – edge flow values
- **failure_df** (*pandas.DataFrame*[72]) – edge failure values
- **tons_col** (*str*[73]) – name of column of tonnages in flow dataframe
- **dist_col** (*str*[74]) – name of column of distance in flow dataframe
- **time_col** (*str*[75]) – name of column of time in flow dataframe
- **cost_col** (*str*[76]) – name of column of cost in flow dataframe
- **vehicle_col** (*str*[77]) – name of column of vehicle counts in flow dataframe
- **changing_tonnages** (*bool*[78]) –

Returns **flow_df_select** – Of edge flow and failure values merged

Return type pandas.DataFrame[79]

network_failure_assembly_shapefiles(*edge_failure_dataframe*, *gdf_edges*, *save_edges=True*, *shape_output_path=''*)

Write results to Shapefiles

Outputs gdf_edges - a Shapefile with results of edge failure dataframe

Parameters

- **edge_failure_dataframe** – Pandas DataFrame of edge failure results
- **gdf_edges** – GeoDataFrame of network edge set with edge ID's and geometry
- **save_edges** (*bool*[80]) – Boolean condition to tell code to save created edge shapefile
- **shape_output_path** (*str*[81]) – Path where the output shapefile will be stored

network_od_path_estimations(*graph*, *source*, *target*, *cost_criteria*, *time_criteria*)

Estimate the paths, distances, times, and costs for given OD pair

Parameters

- **graph** – igraph network structure
- **source** – String/Float/Integer name of Origin node ID
- **source** – String/Float/Integer name of Destination node ID
- **tonnage** (*float*[82]) – value of tonnage
- **vehicle_weight** (*float*[83]) – unit weight of vehicle
- **cost_criteria** (*str*[84]) – name of generalised cost criteria to be used: min_gcost or max_gcost
- **time_criteria** (*str*[85]) – name of time criteria to be used: min_time or max_time

[71] http://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.html#pandas.DataFrame
[72] http://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.html#pandas.DataFrame
[73] https://docs.python.org/3.6/library/stdtypes.html#str
[74] https://docs.python.org/3.6/library/stdtypes.html#str
[75] https://docs.python.org/3.6/library/stdtypes.html#str
[76] https://docs.python.org/3.6/library/stdtypes.html#str
[77] https://docs.python.org/3.6/library/stdtypes.html#str
[78] https://docs.python.org/3.6/library/functions.html#bool
[79] http://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.html#pandas.DataFrame
[80] https://docs.python.org/3.6/library/functions.html#bool
[81] https://docs.python.org/3.6/library/stdtypes.html#str
[82] https://docs.python.org/3.6/library/functions.html#float
[83] https://docs.python.org/3.6/library/functions.html#float
[84] https://docs.python.org/3.6/library/stdtypes.html#str
[85] https://docs.python.org/3.6/library/stdtypes.html#str

- **fixed_cost** (*bool*[86]) –

Returns

- **edge_path_list** (*list[list]*) – nested lists of Strings/Floats/Integers of edge ID's in routes
- **path_dist_list** (*list[float]*) – estimated distances of routes
- **path_time_list** (*list[float]*) – estimated times of routes
- **path_gcost_list** (*list[float]*) – estimated generalised costs of routes

rearrange_minmax_values(*edge_failure_dataframe*)

Write results to Shapefiles

Parameters **edge_failure_dataframe** (*pandas.DataFrame*[87]) – with min-max columns

Returns **edge_failure_dataframe** – With columns where min < max

Return type pandas.DataFrame[88]

spatial_scenario_selection(*network_shapefile*, *polygon_dataframe*, *hazard_dictionary*, *data_dictionary*, *network_id_column*, *network_type='nodes'*)

Intersect network edges/nodes and boundary Polygons to collect boundary and hazard attributes

Parameters

- network_shapefile - Shapefile of edge LineStrings or node Points
- polygon_shapefile - Shapefile of boundary Polygons
- hazard_dictionary - Dictionary of hazard attributes
- data_dictionary - Dictionary of network-hazard-boundary intersection attributes
- network_type - String value -'edges' or 'nodes' - Default = 'nodes'
- name_province - String name of province if needed - Default = ''

Outputs

data_dictionary - Dictionary of network-hazard-boundary intersection attributes:

- edge_id/node_id - String name of intersecting edge ID or node ID
- length - Float length of intersection of edge LineString and hazard Polygon: Only for edges
- province_id - String/Integer ID of Province
- province_name - String name of Province in English
- district_id - String/Integer ID of District
- district_name - String name of District in English
- commune_id - String/Integer ID of Commune
- commune_name - String name of Commune in English
- hazard_attributes - Dictionary of all attributes from hazard dictionary

swap_min_max(*x*, *min_col*, *max_col*)

Swap columns if necessary

write_flow_paths_to_network_files(*save_paths_df*, *min_industry_columns*, *max_industry_columns*, *gdf_edges*, *save_csv=True*, *save_shapes=True*, *shape_output_path=''*, *csv_output_path=''*)

Write results to Shapefiles

[86] https://docs.python.org/3.6/library/functions.html#bool
[87] http://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.html#pandas.DataFrame
[88] http://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.html#pandas.DataFrame

Outputs `gdf_edges` - a shapefile with minimum and maximum tonnage flows of all commodities/industries for each edge of network.

Parameters

- **save_paths_df** – Pandas DataFrame of OD flow paths and their tonnages
- **industry_columns** – List of string names of all OD commodities/industries indentified
- **min_max_exist** – List of string names of commodity/industry columns for which min-max tonnage column names already exist
- **gdf_edges** – GeoDataFrame of network edge set
- **save_csv** – Boolean condition to tell code to save created edge csv file
- **save_shapes** – Boolean condition to tell code to save created edge shapefile
- **shape_output_path** – Path where the output shapefile will be stored
- **csv_output_path** – Path where the output csv file will be stored

1.6.6 atra.utils module

Shared plotting functions

class Style

Bases: `tuple`[89]

Style(color, zindex, label): class to hold an element's styles

Used to generate legend entries, apply uniform style to groups of map elements (See network_map.py for example.)

color
Alias for field number 0

label
Alias for field number 2

zindex
Alias for field number 1

assign_value_in_area_proportions(*poly_1_gpd*, *poly_2_gpd*, *poly_attribute*)

assign_value_in_area_proportions_within_common_region(*poly_1_gpd*, *poly_2_gpd*, *poly_attribute*, *common_region_id*)

count_points_in_polygon(*x*, *points_sindex*)
Count points in a polygon

Parameters

- **x** – row of dataframe
- **points_sindex** – spatial index of dataframe with points in the region to consider

Returns

Return type Number of points in polygon

extract_gdf_values_containing_nodes(*x*, *sindex_input_gdf*, *input_gdf*, *column_name*)

extract_nodes_within_gdf(*x*, *input_nodes*, *column_name*)

extract_value_from_gdf(*row*, *gdf_sindex*, *gdf*, *column_name*)

Inputs are:

[89] https://docs.python.org/3.6/library/stdtypes.html#tuple

row – row of dataframe gdf_sindex – spatial index of dataframe of which we want to extract the value gdf – GeoDataFrame of which we want to extract the value column_name – column that contains the value we want to extract

Outputs are: extracted value from other gdf

gdf_clip (*shape_in*, *clip_geom*)

Inputs are: shape_in – path string to shapefile to be clipped

Outputs are: province_geom – shapely geometry of province for what we do the calculation

gdf_geom_clip (*gdf_in*, *clip_geom*)
Filter a dataframe to contain only features within a clipping geometry

Parameters

- **gdf_in** – geopandas dataframe to be clipped in
- **province_geom** – shapely geometry of province for what we do the calculation

Returns

Return type filtered dataframe

generate_weight_bins (*weights*, *n_steps=9*, *width_step=0.01*, *interpolation='linear'*)
Given a list of weight values, generate <n_steps> bins with a width value to use for plotting e.g. weighted network flow maps.

get_axes (*extent=(-74.04*, *-52.9*, *-20.29*, *-57.38)*, *epsg=None*)
Get map axes

Default to Argentina extent // Lambert Conformal projection

get_data (*filename*)
Read in data (as array) and extent of each raster

get_nearest_node (*x*, *sindex_input_nodes*, *input_nodes*, *id_column*)
Get nearest node in a dataframe

Parameters

- **x** – row of dataframe
- **sindex_nodes** – spatial index of dataframe of nodes in the network
- **nodes** – dataframe of nodes in the network
- **id_column** – name of column of id of closest node

Returns

Return type Nearest node to geometry of row

get_nearest_node_within_region (*x*, *input_nodes*, *id_column*, *region_id*)

legend_from_style_spec (*ax*, *styles*, *loc='lower left'*)
Plot legend

line_length (*line*, *ellipsoid='WGS-84'*)
Length of a line in meters, given in geographic coordinates.

Adapted from https://gis.stackexchange.com/questions/4022/looking-for-a-pythonic-way-to-calculate-the-length-of-a-wkt-li answer-115285

Parameters

- **line** – a shapely LineString object with WGS-84 coordinates.
- **ellipsoid** – string name of an ellipsoid that *geopy* understands (see http://geopy.readthedocs.io/en/latest/#module-geopy.distance).

Returns Length of line in kilometers.

load_config()
Read config.json

load_labels(*data_path*, *include_regions*)

plot_basemap(*ax*, *data_path*, *focus='ARG'*, *neighbours=('CHL', 'BOL', 'PRY', 'BRA', 'URY')*, *country_border='white'*, *plot_regions=True*)
Plot countries and regions background

plot_basemap_labels(*ax*, *data_path*, *labels=None*, *include_regions=False*, *include_zorder=2*)
Plot countries and regions background

round_sf(*x*, *places=1*)
Round number to significant figures

save_fig(*output_filename*)

scale_bar(*ax*, *length=100*, *location=(0.5, 0.05)*, *linewidth=3*)
Draw a scale bar

Adapted from https://stackoverflow.com/questions/32333870/how-can-i-show-a-km-ruler-on-a-cartopy-matplotlib-plot/35705477#35705477

Parameters

- **ax** (*axes*) –
- **length** (*int*[90]) – length of the scalebar in km.
- **location** (*tuple*[91]) – center of the scalebar in axis coordinates (ie. 0.5 is the middle of the plot)
- **linewidth** (*float*[92]) – thickness of the scalebar.

set_ax_bg(*ax*, *color='#c6e0ff'*)
Set axis background color

transform_geo_file(*source_file*, *sink_file*, *sink_schema*, *transform_record*)
Transform a fiona-readable file

Parameters

- **source_file** (*str*[93]) – source file path
- **sink_file** (*str*[94]) – destination file path
- **sink_schema** (*dict*[95]) – fiona schema for output
- **transform_record** (*function*) – function that accepts a fiona record and returns a fiona record or None

voronoi_finite_polygons_2d(*vor*, *radius=None*)
Reconstruct infinite voronoi regions in a 2D diagram to finite regions.

Source: https://stackoverflow.com/questions/36063533/clipping-a-voronoi-diagram-python

Parameters

- **vor** (*Voronoi*) – Input diagram
- **radius** (*float*[96]*, optional*) – Distance to 'points at infinity'

Returns

[90] https://docs.python.org/3.6/library/functions.html#int
[91] https://docs.python.org/3.6/library/stdtypes.html#tuple
[92] https://docs.python.org/3.6/library/functions.html#float
[93] https://docs.python.org/3.6/library/stdtypes.html#str
[94] https://docs.python.org/3.6/library/stdtypes.html#str
[95] https://docs.python.org/3.6/library/stdtypes.html#dict
[96] https://docs.python.org/3.6/library/functions.html#float

- **regions** (*list of tuples*) – Indices of vertices in each revised Voronoi regions.
- **vertices** (*list of tuples*) – Coordinates for revised Voronoi vertices. Same as coordinates of input vertices, with 'points at infinity' appended to the end

within_extent (*x*, *y*, *extent*)
Test x, y coordinates against (xmin, xmax, ymin, ymax) extent

1.7 MIT License

```
MIT License

Copyright (c) 2018 oi-analytics

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.
```

1.8 Developers

- Raghav Pant
- Tom Russell
- Elco Koks
- Roald Schoenmakers

CHAPTER 2

Indexes and tables

- genindex
- modindex
- search

CHAPTER 3

Acknowledgements

This project has been developed by Oxford Infrastructure Analytics as part of a project funded by the World Bank and the Global Facility for Disaster Reduction and Recovery (GFDRR).

[97] https://github.com/oi-analytics/argentina-transport

Python Module Index

a

m

n

p

s

t

u

Index

A

C

D

E

G

H

I

J

L

M

N

P

R

S

T

V

W

Z