**Accompanying code for**

The effects of local interventions on global technological change through spillovers: a modelling framework and application to the road-freight sector

Bessie Noll1\*, Bjarne Steffen2,3, Tobias S. Schmidt1,3

1 Energy and Technology Policy Group, Swiss Federal Institute of Technology, ETH Zurich, Clausiusstrasse 37, 8092 Zurich, Switzerland

2 Climate Finance and Policy Group, Swiss Federal Institute of Technology, ETH Zurich, Clausiusstrasse 37, 8092 Zurich, Switzerland

3 Institute of Science, Technology and Policy, Swiss Federal Institute of Technology, ETH Zurich, 8092 Zurich, Switzerland

**\* Correspondence:** [bessie.noll@gess.ethz.ch](mailto:bessie.noll@gess.ethz.ch)

*Code Written by:* Bessie Noll

*Last Updated:* 22.08.2023

# Getting Started

The model contains multiple interactive Python modules that are imported and referenced in various sub-levels of the outermost main module.

## File Structure

* SD Model – Road Freight
  + data
    - \_01\_Final Output Files
      * \_01\_Output Plots
        + All plots will be saved here (.png).
      * Files containing all main model output data (.xlsx, .pkl).
    - \_02\_Use Case Characterization Output Files
      * Files containing all use case characterization data initialized before the main model run (.xlsx, .pkl). If this folder is empty, the *useCaseCharacterization.py* module must be run fully.
    - Model Architecture Data.xlsx
    - TCO Data.xlsx
    - Control Panel.xlsx
    - \_model\_architecture\_data.xlsx
    - \_tco\_data.xlsx
    - \_control\_panel.xlsx
  + model
    - main.py **(primary run)**
    - technologySelection.py
    - totalCostOfOwnership.py
    - capex.py
    - opex.py
    - externalParameters.py
    - intermediateOutputs.py
    - useCaseCharacterization.py
    - experienceCurves.py
    - plot.py **(secondary run, if desired)**
  + model\_environment.yml (anaconda environment)
  + README.docx
  + AUTHORS.md

## Data Management

* All python module files are stored in the folder called *model*.
* All data files (input and output) are stored in the folder called *data*.
  + Within the *data* folder there are two sub-folders: one for final output data files (*\_01\_Final Output Files*) and another for use case characterization output data files (*\_02\_Use Case Characterization Output Files*). The code will automatically send output files to their respective corresponding folders. Output data files from the useCaseCharacterization.py module, which either runs prior to the main model or is pre-initialized, are sent to *\_02\_Use Case Characterization Output Files.* Output data files from the main model are sent to *\_01\_Final Output Files.* Output plot files are sent to *\_01\_Final Output Files -> \_01\_Output Plots.*
* There are 6 input data files—three user interactive excel files and three linked sister excel files that are read directly into the model. It is important to note that any new data input or data alteration must be entered into the user interactive excel files, which all have capitalized headings and spacing instead of underscores (i.e. ‘Model Architecture Data.xlsx). The linked sister excel file (i.e. ‘\_model\_architecture\_data.xlsx’) will then be automatically updated and must be saved in order for the model to read in the new or altered data input.
* All input data files (the sister files) are imported and stored into the model through the externalParameters.py module.
* Intermediate data generated during model runs are recorded and stored primarily in the intermediateOutputs.py module, though some data are also stored in the experienceCurve.py module as well as the main.py module.

## Running the Main Module

1. Extract the SD\_Model\_Road\_Freight.zip file to a designated path on your computer.
2. Import the anaconda environment file (model\_environment.yml) to your anaconda navigator.
3. Activate in terminal the anaconda environment for correct package dependencies. Activation call: >*conda activate (name of imported anaconda environment)*
4. Navigate in terminal to where the model folder is saved on your computer.
5. Run the main.py module by calling: >*python main.py*

NOTE: All directory path calls from different modules should be automatically initialized as they are linked to your working directory where the model is saved and run.

NOTE: The model is designed to run on a Windows operating system and has not been proofed for other operating systems such as MacOSX.

## Running the Plots Module

Once the main module run has been completed and all output data is stored, the plots.py module can be run using the following run call: >*python plots.py*.

## Initializing the Use Case Characterization Module

The module useCaseCharacterization.py is initialized towards the beginning of the main.py module. The module itself computes use case parameters for all simulated investors for different vehicle technologies in different applications and regions. These parameters include: weight, range, power, energy, payload factor, loading factor and empty run share. Depending on the number of simulated investors (i.e. N=3 investors or N=1000 investors), initializing all use case parameters may take a lot of time. The module therefore gives the option to read in pre-initialized use case parameters from the folder *\_02\_Use Case Characterization Output Files.*

Note, however, that in order to run the pre-initialized use case parameter files, two things must first be checked in the former folder.

1. The number of simulated model investors (see Monte Carlo Simulations under the Control Panel section) must match the number of simulations in the pre-initialized use case parameter file called ‘\_USE\_CASE\_DF\_(*constant/distributed*)\_(*N\_simulations*).pkl’. For example, if the parameter *Monte Carlo Simulations* is set to N=1000 in Control Panel.xlsx/\_Features, there must be a pre-initialized use case parameter file with the same number of simulations.
2. The model has the option to run the use case parameters as *constant* or *distributed* values. Here again, similar to the number of Monte Carlo simulations above, *Use Case Parameters Type*, set in Control Panel.xlsx/\_Features, must match the parameter type in the pre-initialized use case parameter file called ‘\_USE\_CASE\_DF\_(*constant/distributed*)\_(*N\_investors*).pkl’.

If both the number of simulations (i.e. N=100) and the parameter type (i.e. type: distributed) set in Control Panel.xlsx/\_Features match the pre-initialized use case parameters pickle file (i.e. ‘\_USE\_CASE\_DF\_distributed\_100.pkl’), then the ‘Initialize Use Case Parameters Boolean’ feature can be set to False.

# Citation

If you use this model for analysis, the following should be used as citation:

(to be inserted upon publication)

# License

ETH Zurich, Energy and Technology Policy Group (EPG), EU Horizons 2020 Marie Sklodowska-Curie (MSC) grant agreement No. 847585.