This part details the Python codes by Herman utilized for estimating the structural gravity model and obtaining the general equilibrium effects. The gegravity Python package is used for this task due to its comprehensive functionalities tailored for estimating structural gravity models and simulating scenarios for counterfactual analysis. The OneSectorGE model within the gegravity package is an adaptation of the structural model devised by Yotov et al. (2016). The relevant excerpts of the code are provided.

#installation

pip install gegravity

import gegravity as ge

import pandas as pd

# Increase number of columns printed for a pandas DataFrame

pd.set\_option("display.max\_columns", None)

pd.set\_option('display.width', 1000)

import gme as gme

#Load the Gravity data

from google.colab import files

uploaded = files.upload()

gravity\_data\_location = "/content/DEBASHREE\_FINALDATASET\_200307.csv"

grav\_data = pd.read\_csv(gravity\_data\_location)

df = pd.read\_csv('/content/DEBASHREE\_FINALDATASET\_200307.csv')

print(grav\_data.head())

#preparing data for GE model

# Define GME Estimation Data

gme\_data = gme.EstimationData(grav\_data, # Dataset

imp\_var\_name="iso\_d", # Importer column name

exp\_var\_name="iso\_o", # Exporter column name

year\_var\_name = "year", # Year column name

trade\_var\_name="trade") # Trade column name

# Create Gravity Model

gme\_model = gme.EstimationModel(gme\_data, # Specify data to use

lhs\_var="trade", # dependent, "left hand side" variable

rhs\_var=["RTA","contiguity", # independent variables

"comcol","comlang","lndist","INTL\_BRDR"],

fixed\_effects=[["exporter"],["importer"]]) # Fixed effects to use

# Estimate gravity model with PPML

gme\_model.estimate()

# Print econometric results table

print(gme\_model.results\_dict['all'].summary())

# Define GE model

ge\_model = ge.OneSectorGE(gme\_model, # gme gravity model

year = "2021", # Year to use for model

expend\_var\_name = "gdp\_d", # Expenditure column name

output\_var\_name = "gdp\_o", # Output column name

reference\_importer = "IND", # Reference importer

sigma = 5) # Elasticity of substitution

# Test that the model system of equations is computable from the supplied data and parameters

test\_diagnostics = ge\_model.test\_baseline\_mr\_function()

# See what is returned:

print(test\_diagnostics.keys())

# Check the values of the model parameters computed from the baseline data, which should be numeric with no missing values

input\_params = test\_diagnostics['mr\_params']

# Check one set of parameters, for example:

print(input\_params['cost\_exp\_shr'])

# Check for OMR rescale factors that results in convergence

rescale\_eval = ge\_model.check\_omr\_rescale(omr\_rescale\_range=3)

print(rescale\_eval)

# Solve the baseline model

ge\_model.build\_baseline(omr\_rescale=100)

# Examine the solutions for the baseline multilateral resistances

print(ge\_model.baseline\_mr.head(12))

# Create a copy of the baseline data

exp\_data = ge\_model.baseline\_data.copy()

#We develop an array of all member countries of BRICS.

member\_countries = ["BRA", "RUS", "IND", "CHN", "ZAF"]

#For our counterfactual experiment, we set the value of the PTA dummy variable with all the member countries as 0. We use 'for' loop for this iterative purpose.

for country in member\_countries:

exp\_data.loc[(exp\_data["iso3\_d"] == "IND") | (exp\_data["iso3\_o"] == country), "pta"] = 0

exp\_data.loc[(exp\_data["iso3\_d"] == country) | (exp\_data["iso3\_o"] == "IND"), "pta"] = 0

# Define the experiment within the GE model

ge\_model.define\_experiment(exp\_data)

# Examine the baseline and counterfactual trade costs

print(ge\_model.bilateral\_costs.head(56))

# Simulate the counterfactual model

ge\_model.simulate()

# We can examine the counterfactual trade flows predicted by the model.

print(ge\_model.bilateral\_trade\_results.head())

# A collection of many of the key country-level results (prices, total imports/exports, GDP, welfare, etc.)

country\_results = ge\_model.country\_results

print(country\_results)

# The bilateral trade results

bilateral\_results = ge\_model.bilateral\_trade\_results

print(bilateral\_results)

# A wider selection of aggregate, country-level trade results

agg\_trade = ge\_model.aggregate\_trade\_results

print(agg\_trade)

# country multilateral resistance (MR) terms

mr\_terms = ge\_model.country\_mr\_terms

# Get the solver diaganoistics, which is a dictionary containing many types of solver diagnostic info

solver\_diagnostics = ge\_model.solver\_diagnostics

mr\_terms = ge\_model.country\_mr\_terms

print(mr\_terms)

solver\_diagnostics = ge\_model.solver\_diagnostics

# Export the results to a collection of spreadsheet (.csv) files and add trade values in levels to the outputs.

from google.colab import drive

drive.mount('/content/drive')

import os

output\_directory = "/content/drive/My Drive/"

os.makedirs(output\_directory, exist\_ok=True)

ge\_model.export\_results(directory=output\_directory, name="GE\_analysis1")

#For our counterfactual experiment, we set the value of PTA dummy variable with all the member countries as 0. We use 'for' loop for this iterative purpose.

for country in member\_countries:

exp\_data.loc[(exp\_data["iso3\_d"] == "CHN") | (exp\_data["iso3\_o"] == country), "pta"] = 0

exp\_data.loc[(exp\_data["iso3\_d"] == country) | (exp\_data["iso3\_o"] == "CHN"), "pta"] = 0

# Define the experiment within the GE model

ge\_model.define\_experiment(exp\_data)

# Examine the baseline and counterfactual trade costs

print(ge\_model.bilateral\_costs.head(56))

# Simulate the counterfactual model

ge\_model.simulate()

# We can examine the counterfactual trade flows predicted by the model.

print(ge\_model.bilateral\_trade\_results.head())

# A collection of many of the key country-level results (prices, total imports/exports, GDP, welfare, etc.)

country\_results = ge\_model.country\_results

print(country\_results)

# The bilateral trade results

bilateral\_results = ge\_model.bilateral\_trade\_results

print(bilateral\_results)

# A wider selection of aggregate, country-level trade results

agg\_trade = ge\_model.aggregate\_trade\_results

print(agg\_trade)

# country multilateral resistance (MR) terms

mr\_terms = ge\_model.country\_mr\_terms

# Get the solver diaganoistics, which is a dictionary containing many types of solver diagnostic info

solver\_diagnostics = ge\_model.solver\_diagnostics

mr\_terms = ge\_model.country\_mr\_terms

print(mr\_terms)

solver\_diagnostics = ge\_model.solver\_diagnostics

# Export the results to a collection of spreadsheet (.csv) files and add trade values in levels to the outputs.

from google.colab import drive

drive.mount('/content/drive')

import os

output\_directory = "/content/drive/My Drive/"

os.makedirs(output\_directory, exist\_ok=True)

ge\_model.export\_results(directory=output\_directory, name="GE\_analysis2")

**Python Code File**

**India exit -**

[**Codes**](https://colab.research.google.com/drive/17_DHx3BBLqG3H_Ls9u9PN7CnWr5vcmK-#scrollTo=2zAtFcGALrve)

[**Results**](https://drive.google.com/drive/folders/1CzKRATdv3Y-jfquzVIBdy6MmCeRYsfIz?usp=drive_link)

**China Exit-**

[**Codes**](https://colab.research.google.com/drive/1XJfSH9ECQNdmmyi93v5Blq96LKwsMZsg#scrollTo=6WRwSjMZNWQt)

[**Results**](https://drive.google.com/drive/folders/1W7sTXsFtd4YaCyIgP0fb_y5zMgOdnhJp?usp=drive_link)