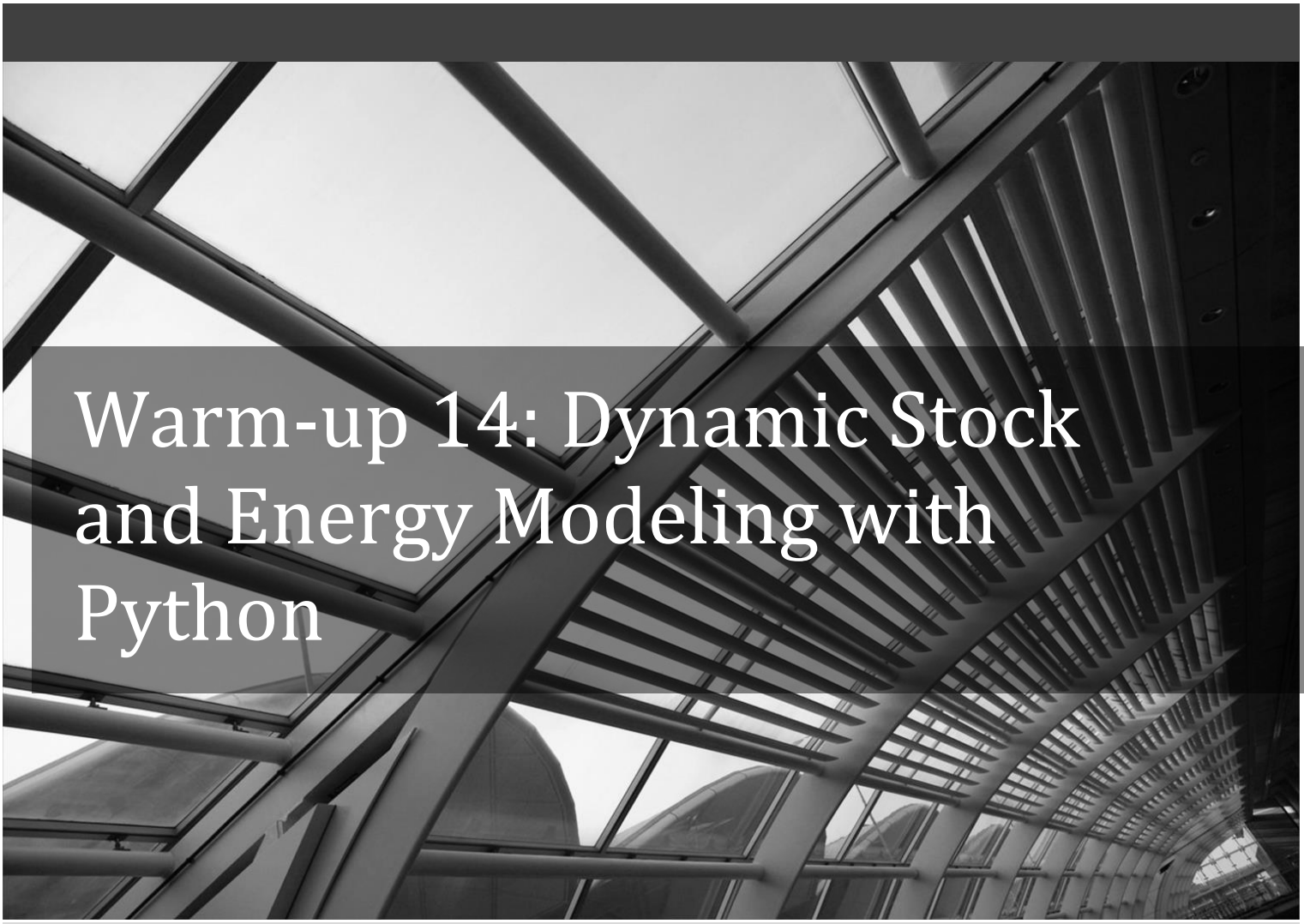

Warm-up 14: Dynamic Stock and Energy Modeling with Python

Assignment

TEP4290 – Modelling of Built Environment System
Spring 2024

Prof. Dr. Daniel B. Müller
PhD Cand. Nils Dittrich
MSc Zoé Cord´Homme

INDUSTRIAL ECOLOGY PROGRAMME

Warm-up 14: Dynamic Stock and Energy Modelling with Python

Introduction

Transportation is a major contributor to climate change. In the attached journal paper (Pauliuk et al., 2012), we performed a case study on the possible future development of the Chinese passenger vehicle fleet and its carbon footprint using a dynamic stock model.

The purpose of this exercise is to make you familiar with the problem setting, the state-of-the-art modeling of dynamic stocks, the relationship between stocks and energy consumption, and how dynamic stock models can be programmed in Python. Using the passenger car stock in China as an example you shall learn how a stock can be estimated if the *inflow and the lifetime are known* (**inflow-driven model**), and how one can derive future inflows and outflows from scenario *assumptions on how the stock will develop* over time (**stock-driven model**). Finally, you are asked to determine the total kilometers driven by the entire stock for each year and to derive the associated direct carbon emissions from gasoline combustion.

To apply the lifetime model one needs to track each cohort in the stock separately. A stock with a known subdivision into cohorts can be represented as a time-cohort matrix (Fig. 1) where each cohort is tracked over time. In the example below, the product lifetime is five years sharp. The time-cohort matrix contains the full information about the stock: the total stock, inflow, and outflow for each year can be derived from it.

Cohort→ Time↓	1	2	3	4	5	6	7	8	9	10
1	10	0	0	0	0	0	0	0	0	0
2	10	12	0	0	0	0	0	0	0	0
3	10	12	7	0	0	0	0	0	0	0
4	10	12	7	9	0	0	0	0	0	0
5	10	12	7	9	10	0	0	0	0	0
6	0	12	7	9	10	12	0	0	0	0
7	0	0	7	9	10	12	8	0	0	0
8	0	0	0	9	10	12	8	14	0	0
9	0	0	0	0	10	12	8	14	9	0
10	0	0	0	0	10	12	8	14	9	10

Figure 1. Example of a time-cohort matrix for a lifetime of five years sharp

Preparation

1. Read and understand the work presented in the article (Pauliuk et al., 2012). Reading the paper's Supplementary Information is highly recommended as it presents a more comprehensive mathematical description of the model.

2. Read and understand the dynamic_stock_model package documentation:

   https://github.com/stefanpauliuk/dynamic_stock_model

Coding instructions

From 1950 to 2008, we know the population and how many new cars have been registered (the inflow). Your task is to determine the cars that have already left the use phase and the size of the car stock, using a normally distributed passenger car lifetime. From 2009 on, we have a population projection from the UN and a scenario curve for the number of passenger cars per 1000 inhabitants. The end value of 450 in 2050 is typical of European countries. Here, from 2009 on, your task is to determine the required inflow of cars and how many cars are leaving use. This is done in a year-by-year calculation, starting in 2009, where we know the age structure of the existing stock.

Name the parameters and variables with a subscript, e.g. "_t" or "_tc", to indicate the dimensions of the matrix. "stock_t" is a vector with the total stock through time and length equal to the number of years (t), while "stock_tc" is a matrix with time (t) as the first dimension and cohort (c) as the second dimension. This makes it easier to keep track of the exact content of a variable and its size.

You are free to use the dynamic_stock_package or to make your own functions using for loops, the *norm* function from the package *scipy,* and array operators from the *numpy* package.

The following lifetime distribution λ shall be used:

$$\lambda(t,c,\tau,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(t-c-\tau)^2}{2\sigma^2}\right)$$

Equation 1

where,

t: current year, e.g. 2016

c: cohort, e.g. 1996

τ: mean lifetime, use 15 years

σ: width of distribution, standard deviation, use 5 years

Only the difference t-c varies in the equation, which means that the probability of a car being scrapped is determined by its age.

All your script should be written in the attached Jupyter notebook, that you are free to modify for this.

Tasks

1. Import the necessary packages and read in the given data (Excel file *"Exercise_1_Data.xls"*) to the jupyter notebook.

2. Determine the time-cohort matrix for the years **1950 - 2008** (**inflow-driven model**). Then determine the outflow, stock change, stock, and number of cars per 1000 people from the time-cohort matrix and population change.
3. Continue working on your script and complete the time-cohort matrix and compute the total stock, stock change, inflow, and outflow for the years **2009 - 2050** using the per capita car stock and the population as the model driver (**stock-driven model**).

4. Also in your script, determine the total number of kilometers driven, the total gasoline consumption, and the total direct carbon emissions for all years.

5. Generate and export the following plots: (i) Inflow and Outflow, (ii) Stock, (iii) Stock Change, (iv) Total km driven, (v) Total gasoline consumed, and (vi) Total direct CO_2 emissions.

6. Export all your results to the Excel file *"Exercise_1_Data.xls"*. Fill out the blanks in the table on the sheet *'CarStockData'*, and write the time-cohort matrix on the sheet *'Stock_TC'*.

7. Interpret your findings. By how much will the total kilometrage and the total carbon footprint increase between 2000 and 2050? What about the life-cycle footprint of a single car?

Deliverables and Deadline

Submit the excel file with the completed tables, and the Jupyter notebook containing all your code. The notebook shall contain the entire computation chain: data reading and formatting, computing, plotting, exporting of figures, and writing of the results. Structure your code, use understandable variable names, and comment your code well! Please make use of the Markdown cells for writing longer sections of text.

The deadline for submission is February 6th @ 18:00h on Blackboard.
This exercise is individual work, you are not allowed to use a group submission.

Make use of the exercise hours and the discussion board if things do not work as they should. However, try to use the Python help and the web (Google, stackoverflow.com) first.