Monte Carlo Simulation

Amina Abdelhedi

1 Introduction

To build a Life Cycle Assessment (LCA), we need to look at the "Use Phase." This part of a product's life is a major part of its carbon footprint, but it is hard to measure with one fixed number because it depends on how different people behave. In this case, we model the Use Phase as a shower event.

We use a Monte Carlo simulation to get a better view of the results. By using probability distributions for the main variables instead of simple averages, we can see the range of carbon emissions from the best to the worst cases.

2 Methodology and Model

2.1 Carbon Emission Model

The total carbon footprint of a shower (C_{total}) is calculated using the equation below.

$$C_{total} = (L \times F) \times [EF_{water} + (E_{heat} \times EF_{energy})] \tag{1}$$

Where:

- L: Shower duration (minutes), follows a lognormal distribution (mean = 10, standard deviation = 3).
- F: Water flow rate (liters per minute), fixed value = 0.144 .
- EF_{water}: Emission factor for water supply (kg CO_2e/L), modeled using a multinomial distribution based on proportions provided in a dedicated table.
- E_{heat}: Energy required to heat water (MJ/L), modeled using a multinomial distribution based on proportions provided in a dedicated table.
- EF_{energy}: Emission factor for thermal energy (kg CO_2e/MJ).

2.2 Variable Input Parameters simulation

2.2.1 Shower duration

The shower length follows a lognormal distribution with a mean (mean = 10) and a standard deviation (sigma = 3). To simulate 1,000 scenarios, we must determine the distribution parameters using the following two equations:

$$\mu = \ln\big(E(X)\big) - \frac{1}{2}\sigma^2 \tag{2}$$

$$\sigma^2 = \ln\left(1 + \frac{\text{Var}(X)}{E(X)^2}\right) \tag{3}$$

Where:

- X: Shower length (minutes)
- $E(X)$: Mean = 10 minutes
- $V(X)$: Variance = (Standard-deviation/ sigma)^2 = 9

To determine the parameters of the lognormal distribution and perform the simulation, the Python code shown below was used.

```python
import numpy as np

# Given values
E = 10      # Expected value / mean E(X)
Var = 3**2   # Variance Var(X) = sigma **2
# Compute sigma^2 and sigma
sigma2 = np.log(1 + Var / (E**2))
sigma = np.sqrt(sigma2)

# Compute mu
mu = np.log(E) - 0.5 * sigma2
print(mu, sigma)
shower_length = np.random.lognormal(mu, sigma, size=n_sim)
```

Figure 1: Python code to determinate lognormal parameters

The histogram below shows the distribution of the shower duration.

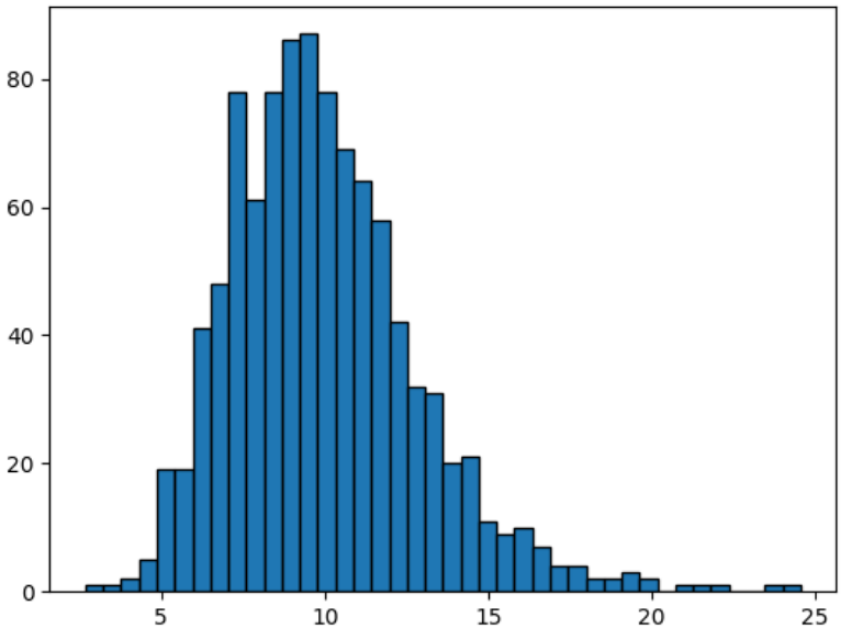

Figure 2: Shower duration histogram

2.2.2 Emission factor - water & Heat energy

Step 1: Data importing and cleaning After importing the initial datasets, we removed empty columns and rows, as well as entries that did not contribute meaningful information to the simulation. This resulted in a cleaner dataset and faster data manipulation.

a. Water EF

```
EF_water2 = EF_water.drop(columns=["ecoinvent-v3.5 Dataset", "Amount (MJ)", "Unit","Comment"])
EF_water2
```

	Water_type	Proportion	CO2	GEO
0	tap water production, underground water with d...	14.29%	0.000136	Europe without Switzerland (Europe without Swi...
1	tap water production, underground water with c...	14.29%	0.000297	Europe without Switzerland (Europe without Swi...
2	tap water production, underground water withou...	14.29%	0.000014	Switzerland (CH)
3	tap water production, conventional treatment	14.29%	0.000218	Europe without Switzerland (Europe without Swi...
4	tap water production, microstrainer treatment	14.29%	0.000376	Rest-of-World (RoW)
5	tap water production, conventional with biolog...	14.29%	0.000288	Europe without Switzerland (Europe without Swi...
6	tap water production, seawater reverse osmosis...	14.29%	0.005770	Global (GLO)

Figure 3: Water EF cleaned table

b. Heat Energy

```
Heat_energy2 = Heat_energy.drop(columns=["ecoinvent-v3.5 Dataset", "Unit","Amount (MJ)"])
Heat_energy2.columns = Heat_energy2.columns.str.replace('\n', '', regex=False).str.strip()
Heat_energy2 = Heat_energy2.iloc[2:].reset_index(drop=True)
Heat_energy2
```

	Heat_type	Comment	Proportion	CO2	GEO
0	Heat, central or small-scale, natural gas {CH}	Share of gas heating	38.55%	0.074034	Europe without Switzerland (Europe without Swi...
1	Heat, central or small-scale, other than natur...	Share of oil heating	21.02%	0.104351	Europe without Switzerland (Europe without Swi...
2	Heat, district or industrial, other than natur...	Share of district heating	12.21%	0.000260	Switzerland (CH)
3	Heat, borehole heat pump {CH}	Share from heat pumps	4.62%	0.029317	Europe without Switzerland (Europe without Swi...
4	Heat, central or small-scale, other than natur...	Share from wood heating	0.73%	0.006018	Switzerland (CH)
5	Heat, central or small-scale, other than natur...	Share from solar thermal	3.71%	0.002794	Switzerland (CH)
6	Electricity, low voltage, Zurich, Switzerland	Share from electricity	19.16%	0.033284	Switzerland (CH)

Figure 4: Heat energy cleaned table

Step 2: Data visualization and understanding To better understand the variability of water and heat energy CO emission factors, the bar charts below were created.

a. Water EF

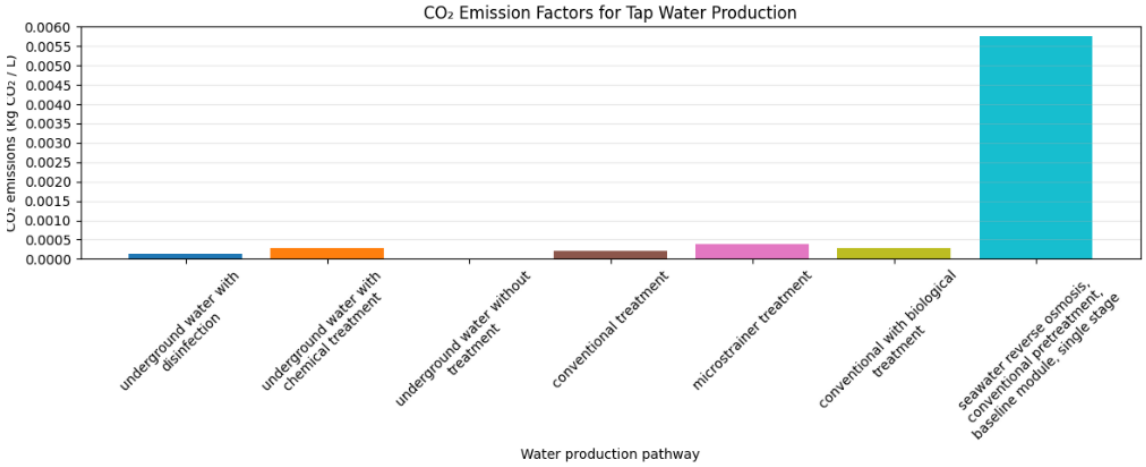

Figure 5: CO Emission Factors for Tap Water Production

b. Heat energy EF

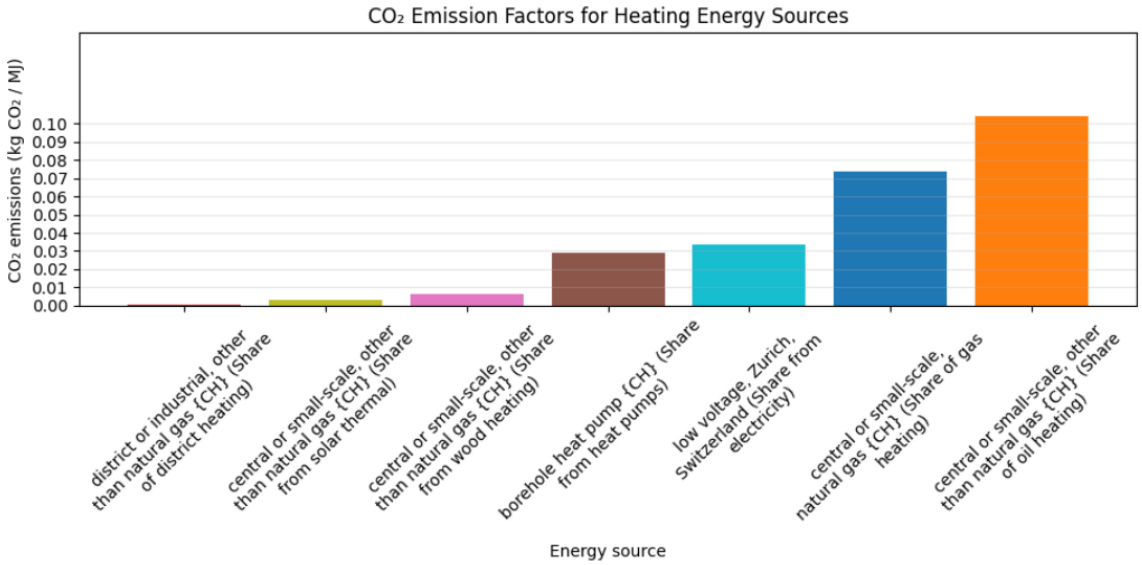

Figure 6: CO Emission Factors for Heating Energy Sources

Step 3: Heat energy simulation) To simulate the heat energy emission factor values, we used the code below.

```
heat = Heat_energy2.copy()

heat_probs = heat["Proportion"].astype(str).str.rstrip("%").astype(float)
heat_probs = heat_probs / heat_probs.sum()

heat_idx = np.random.choice(
    heat.index.values,     # choose among row IDs
    size=n_sim,
    p=heat_probs.values
)

sampled_heat = heat.loc[heat_idx, ["Heat_type", "Comment", "CO2"]]
sampled_heat.head()
```

	Heat_type	Comment	CO2
0	Heat, central or small-scale, natural gas {CH}	Share of gas heating	0.074034
2	Heat, district or industrial, other than natur...	Share of district heating	0.000260
1	Heat, central or small-scale, other than natur...	Share of oil heating	0.104351
6	Electricity, low voltage, Zurich, Switzerland	Share from electricity	0.033284
0	Heat, central or small-scale, natural gas {CH}	Share of gas heating	0.074034

Figure 7: Heat energy simulations

To verify that the simulation respects the given proportions, the pie chart below is used to represent the percentage of each category.

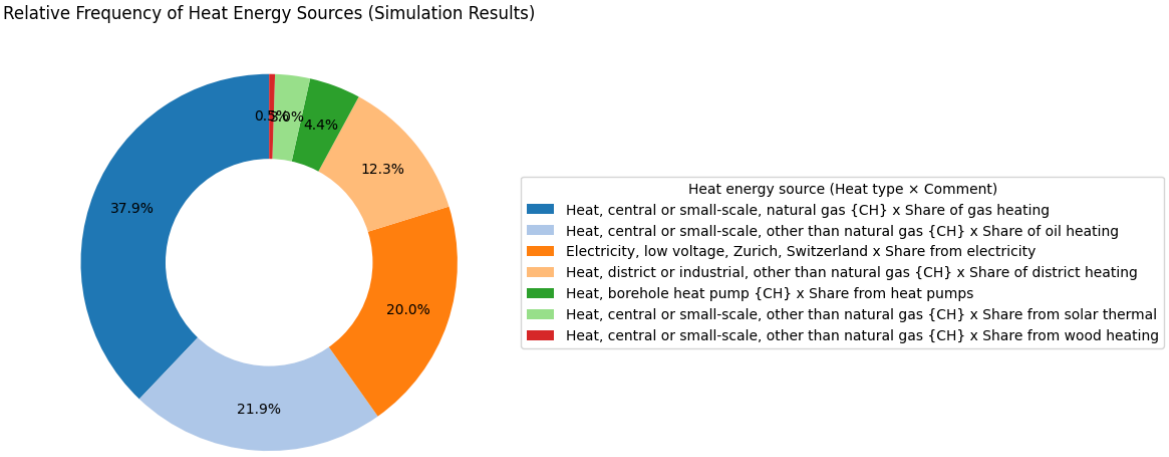

Figure 8: Relative Frequency of Heat Energy Sources (Simulation Results)

Step 4: Water type simulation) To simulate the Water emission factor values, we used the code below.

```
water = EF_water2.copy()

water_probs = water["Proportion"].astype(str).str.rstrip("%").astype(float)
water_probs = water_probs / water_probs.sum()

print(water_probs)

water_idx = np.random.choice(
    water.index.values,     # choose among row IDs
    size=n_sim,
    p=water_probs.values
)

sampled_water = water.loc[water_idx, ["Water_type", "CO2"]]
sampled_water.head()
```

```
0    0.142857
1    0.142857
2    0.142857
3    0.142857
4    0.142857
5    0.142857
6    0.142857
Name: Proportion, dtype: float64
```

	Water_type	CO2
1	tap water production, underground water with c...	0.000297
3	tap water production, conventional treatment	0.000218
2	tap water production, underground water withou...	0.000014

Figure 9: Water EF simulations

To verify that the simulation respects the given proportions, the pie chart below is used to represent the percentage of each category.

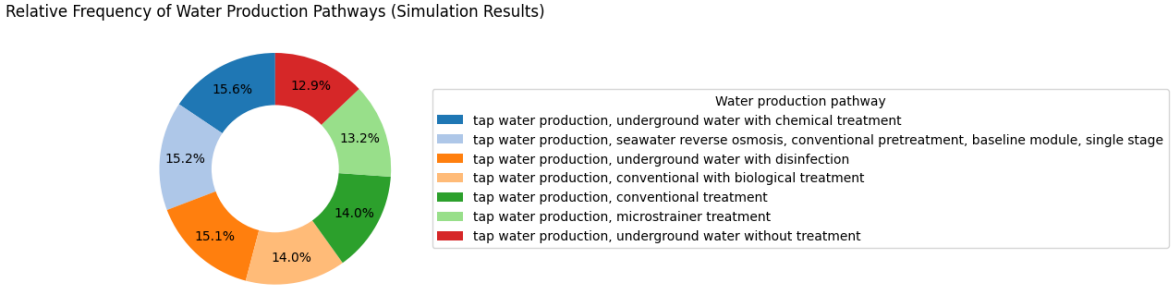

Figure 10: Relative Frequency of Water Production Pathways (Simulation Results)

Step 5 : Total CO2 calculations

Finally, to obtain the resulting table, we used the code below.

```
# --- 2) Make sure sampled tables align (same length and clean index) ---
sampled_water = sampled_water.reset_index(drop=True)
sampled_heat  = sampled_heat.reset_index(drop=True)


heat_label = sampled_heat["Heat_type"].astype(str)
water_label = sampled_water["Water_type"].astype(str)

# --- 3) Compute totals ---
total_water = shower_length * FLOW_RATE_LPM                # Liters
total_heat  = total_water * HEAT_ENERGY_PER_LITER          # MJ

EF_water = sampled_water["CO2"].astype(float)              # kg CO2 / L
EF_heat  = sampled_heat["CO2"].astype(float)               # kg CO2 / MJ

total_CO2eq = total_water * (EF_water + HEAT_ENERGY_PER_LITER * EF_heat)

# --- 4) Build final table ---
results = pd.DataFrame({
    "simulation_number": np.arange(1, n_sim + 1),
    "shower_length": shower_length,
    "total_water": total_water,
    "water_type": sampled_water["Water_type"].astype(str),
    "heat_type": heat_label,          # now only Heat_type
    "total_heat": total_heat,
    "total_CO2eq": total_CO2eq
})

results.head()
```

	simulation_number	shower_length	total_water	water_type	heat_type	total_heat	total_CO2eq
0	1	15.570736	176.964921	tap water production, underground water with c...	Heat, central or small-scale, natural gas {CH}	25.482949	1.939247

Figure 11: Total CO2 table

3 Summary statistics of simulated carbon footprint results

```
results["total_CO2eq"].describe()
```

	total_CO2eq
count	1000.000000
mean	1.064634
std	0.699344
min	0.002797
25%	0.558182
50%	1.040174
75%	1.511444
max	5.195052

Figure 12: CO2 summary report

```
# Boxplot
plt.figure()
plt.boxplot(results["total_CO2eq"], vert=True)
plt.ylabel("Total CO₂eq (kg)")
plt.title("Boxplot of Total CO₂eq per Shower Simulation")
plt.show()
```

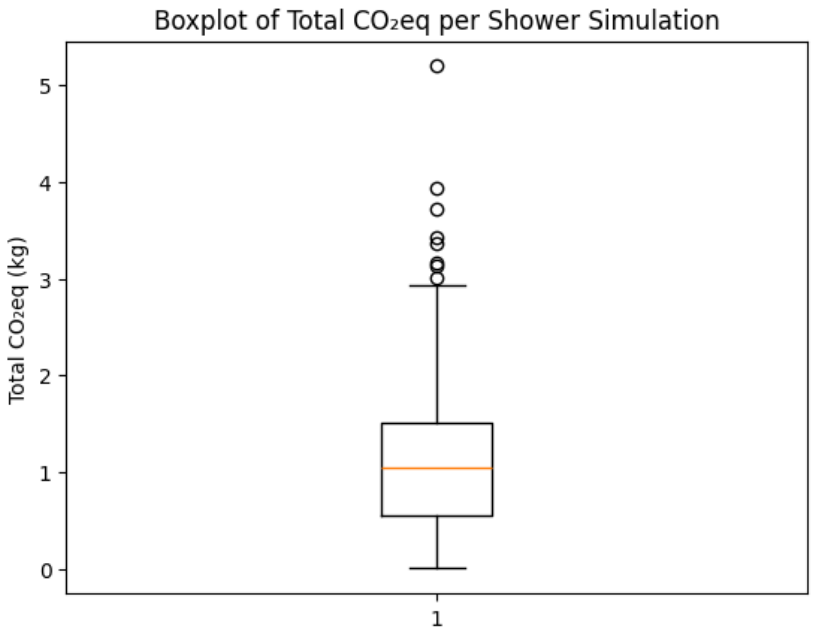

Figure 13: Total CO2 boxplot