

# Simulation Exercises – Parts 1 & 2: Parametric and Multi-Parameter EnergyPlus Automation

If you are using VS Code, press **Ctrl + Shift + V** on Windows or **Cmd + Shift + V** on Mac to open the Markdown Preview panel.

## Overview

- **Part 1 – Parametric Simulation Automation:** Automate the running of pre-built models (Week 2) under different climates using Python and Eppy.
- **Part 2 – Multi-Parameter Dataset Construction:** Expand your script to include envelope and glazing parameters (R-value, WWR, SHGC) across multiple climates.

Together, these two parts prepare you to construct reproducible simulation datasets suitable for analysis and machine learning applications later in the course.

## Part 1 – Parametric Simulation Automation for Different Climates

### Objective

Your task is to **automatically run the previously created models from Week 2 (models 1, 2, and 3)** under different climates using the **Eppy** library.

You will complete a short Python script that automates **multiple EnergyPlus runs** based on a parameter table.

The goal is to fill in and debug the missing parts so that the notebook:

1. Reads case information from a CSV file,
2. Runs each case automatically, and
3. Saves total heating, cooling, and lighting energy use into a summary file.

## Two Versions of the Script


To accommodate different learning needs, you are provided with **two versions** of the exercise script:

Version	Description	Recommended For
<b>Version A – Minimal</b>	Contains only the basic structure and key comments (no working code). You will need to write the full loop logic and file handling steps yourself.	Students who want full practice implementing everything from scratch.
<b>Version B – Guided</b>	Includes most of the code structure, variable setup, and partial logic. You will only need to complete specific missing lines (e.g., file paths, meter names, data assignments).	Students who prefer focusing on understanding simulation flow and debugging rather than writing every line.

Both scripts are functionally equivalent once completed. You are encouraged to start from Version A - Minimal.

## How to Use the Templates

1. Open the chosen script or notebook ( `Part1_minimal.ipynb` or `Part1_guided.ipynb` ) in VS Code.
2. Carefully read the comments marked with `# TODO:` or `# complete this part` .
3. Fill in the missing pieces based on the instructions provided in this document.
4. Test your code with one or two rows from the parametric table before running all simulations.

 *Tip:* Even if you use the guided version, review each line and make sure you understand why it's written that way. The same workflow will be reused in future exercises (e.g., for control logic and multi-zone models).

## Setup Checklist

Before running anything, make sure you have:

- A working EnergyPlus installation ( `energyplus` command runs from terminal).
- A Python environment with `pandas` and `eppy` installed.
- A folder that contains:

```
root_dir/
├─ param_table.csv
├─ model_1.idf, model_2.idf, model_3.idf
└─ Part1_minimal.ipynb, Part1_guided.ipynb
```

## Your Coding Task

The provided skeleton defines the structure of a batch run loop.

You need to ensure the following steps are implemented correctly:

### 1. Read input data

- Load `param_table.csv` and confirm it contains `sim_id`, `weather`, and `model` columns.
- Inspect the first few rows with `.head()` to verify.

### 2. Link to weather files

- Use the provided dictionary (e.g., `MAP_WEATHER`) that maps weather codes in the CSV to actual `.epw` paths.
- Use relative or absolute paths consistently.

### 3. Set up the simulation loop

- For each row, point to the correct IDF and EPW.
- Use `os.path.join()` to build file paths safely.
- Make sure new simulation folders (e.g., `param_sim/sim_1`) are created before running EnergyPlus.

### 4. Run simulations

- Use `idf.run(expandobjects=True)` or equivalent, and watch for errors in the console. Refer to Eppy's Documentation <https://eppy.readthedocs.io/en/latest/index.html>
- If EnergyPlus isn't found, verify your `PATH` or use a full executable path.

### 5. Extract results

- Read the generated `eplusout.eso`.
- Compute total kWh for heating, cooling, and lighting.
- Add these values back into your results DataFrame.

### 6. Save the results

- When the loop finishes, export to `part1_results.csv`.
- Inspect it in Excel or pandas to confirm values make sense.

## High-Level Tips

Stage	Tip
File Paths	Always print or log paths ( <code>print(idf_dir)</code> ) before running — most errors come from wrong directories.
Weather Mapping	Start with 1–2 cases and verify each weather key resolves to a valid <code>.epw</code> file.
EnergyPlus Version	The <code>.idf</code> files must match your installed EnergyPlus version — run one manually first.
Output Meters	Use the <b>exact meter names</b> that exist in your model; spelling and case matter.
Debugging	Wrap risky lines with <code>try/except</code> and print the simulation ID that failed.
Testing	Limit your loop to <code>param_table.head(2)</code> until it works; then expand.
Re-runs	Delete or rename old <code>param_sim</code> folders if EnergyPlus keeps reusing cached results.
Sanity Check	Compare heating/cooling magnitudes across cases — they should reflect climate or model differences.

## Troubleshooting

- **Nothing happens:** EnergyPlus might not be in your PATH — add it or call via its full path.
- **Simulation failed:** Open `epplusout.err` inside the run folder to find the issue.
- **Empty CSV:** Check that your ESO file exists and that the meter names match.
- **Permission errors:** Avoid running in protected directories (e.g., `C:\Program Files` ).

## Deliverables

When finished, save `part1_results.csv`

# Part 2 – Multi-Parameter Dataset Construction

## Objective

Build upon Part 1 by exploring how **multiple envelope and glazing parameters** influence building energy use. You'll automate a series of EnergyPlus runs varying:

- **Climate (4 locations)**
- **Wall insulation level (3 R-values)**
- **Window-to-Wall Ratio (WWR, 3 levels)**
- **Solar Heat-Gain Coefficient (SHGC, 3 levels)**

This exercise produces a larger dataset suitable for model training.

## Simulation Settings

Parameter	Description	Values
Location	Climate zones representing different U.S. cities	San Francisco, Sacramento, Chicago, New York
Wall R-value	Wall insulation level (ft <sup>2</sup> ·°F·hr/Btu)	14.3, 21.9, 29.7
WWR	Window-to-Wall Ratio	0.25, 0.40, 0.60
SHGC	Solar Heat Gain Coefficient	0.25, 0.40, 0.60

 Total simulation cases:  $4 \times 3 \times 3 \times 3 = 108$  runs

## Setup Checklist

- Confirm **EnergyPlus ≥ v9.x** is installed.
- Install dependencies if missing:

```
pip install eppy esoreader tqdm shutil
```

- Verify folder structure:

```
root_dir/
├─ model_week4.idf
└─ Part2_minimal.ipynb, Part2_guided.ipynb
```

## About the Base Model ( model\_week4.idf )

The `model_week4.idf` is a simplified derivative of `model_1.idf` , where both the **wall** and **window** systems were replaced with **simple constructions**.

This simplification makes it easier to modify parameters such as **R-value**, **SHGC**, and **WWR**, enabling efficient batch simulations for parametric analysis.

Such simplifications are acceptable for **whole-building, annual energy simulations**.

At this level, the goal is to explore comparative trends rather than represent full construction detail.

 Note: This setup is intentionally simplified.

For real-world EnergyPlus studies or calibration, the **model detail should align with the simulation objectives** and, where possible, be **calibrated with measured data**.

## (Optional) Example Comparison

**Original ( model\_1.idf ):**

```
Construction,
    exterior wall,
    Brick,
    Insulation,
    Concrete;
```

```
Construction,
    windows,
    CLEAR 6MM,
    AIR 6MM,
    CLEAR 6MM;
```

**Simplified ( model\_week4.idf ):**

```

Material,
    Simple Wall,          !- Name
    Rough,               !- Roughness
    0.2,                 !- Thickness {m}
    0.045,               !- Conductivity {W/m-K}
    2000,                !- Density {kg/m3}
    1000,                !- Specific Heat {J/kg-K}
    0.9,                 !- Thermal Absorptance
    0.7,                 !- Solar Absorptance
    0.7;                 !- Visible Absorptance

WindowMaterial:SimpleGlazingSystem,
    Simple Window,       !- Name
    3.058,               !- U-Factor {W/m2-K}
    0.7,                 !- Solar Heat Gain Coefficient
    0.781;               !- Visible Transmittance

Construction,
    exterior wall,       !- Name
    Simple Wall;         !- Outside Layer

Construction,
    windows,             !- Name
    Simple Window;       !- Outside Layer

```

💡 This streamlined setup keeps the model flexible for parameter sweeps while maintaining realistic energy behavior at the building scale.

## Your Coding Tasks

The provided notebook outlines the structure for a **multi-parameter EnergyPlus batch simulation**. Your task is to complete the missing parts and ensure the workflow runs end-to-end.

### 1. Define parameter mappings

- Fill in `MAP_WEATHER` with correct paths to `.epw` weather files for each city.
- Complete `MAP_WWR` to control window geometry based on the selected WWR.
- Verify units and ensure consistent paths relative to `root_dir`.

### 2. Construct the parameter space

- Use `itertools.product()` to generate all combinations of:  
Location × Wall\_Rvalue × WWR × SHGC .
- Save the resulting DataFrame as `week4_all_parameters.csv` .
- Convert wall R-values from  $\text{ft}^2 \cdot ^\circ\text{F} \cdot \text{hr} / \text{Btu}$   $\rightarrow$   $\text{m}^2 \cdot \text{K} / \text{W}$  before using them.

### 3. Implement the model modification function

- In `modify_idf(idf, modify_dict)` , edit the IDF to reflect current parameters:
  - Update wall conductivity for the given R-value.
  - Adjust window geometry based on WWR.
  - Apply SHGC changes to the glazing layer.
  - Switch to the correct weather file.

### 4. Run batch simulations

- Loop through all parameter combinations in the DataFrame.
- For each iteration:
  - Create a unique output folder (e.g., `run_001` , `run_002` , ...).
  - Call `modify_idf()` and run the simulation.
  - Extract cooling energy from the results.
- Use `tqdm` to visualize progress and wrap the loop in `try/except` to skip failed runs.

### 5. Store and export results

- Append results (parameters + outputs) into a single DataFrame.
- Save to `week4_results.csv` .
- Double-check that the file contains 108 entries and reasonable energy values.

## High-Level Tips

Stage	Tip
<b>Parameter Setup</b>	Test only a few parameter combinations first (e.g., <code>df.head(2)</code> ) before running all 108 cases.
<b>File Paths</b>	Use absolute paths for both the EnergyPlus executable and weather files; print them to verify before batch runs.
<b>Weather Mapping</b>	Double-check each city's <code>.epw</code> file mapping — a missing path will crash the run early.
<b>Unit Conversion</b>	Convert wall R-values to SI units: $1 \text{ ft}^2 \cdot ^\circ\text{F} \cdot \text{hr} / \text{Btu} \approx 0.17611 \text{ m}^2 \cdot \text{K} / \text{W}$ .
<b>Model Editing</b>	Make incremental changes — test your <code>modify_idf()</code> logic for one case before automating all.



Stage	Tip
Simulation Loop	Wrap each iteration in <code>try/except</code> to continue if a case fails; log the failed index for review.
Output Management	Store each run in its own subfolder (e.g., <code>run_001</code> , <code>run_002</code> , ...) to prevent overwriting results.
Performance	EnergyPlus runs can take time — use <code>tqdm</code> for progress visibility and avoid unnecessary reruns.
Sanity Check	After runs, compare total cooling energy across climates — warmer cities (e.g., Sacramento) should show higher values.
Cleanup	Use <code>shutil.rmtree()</code> to remove temp folders after confirming all results are safely saved.

## Output Management Tips

The simulation loop manages **hundreds of temporary folders and output files**, so watch how results are stored and deleted:

```
for groupname in trange(len(df)):
    modify_dict = df.iloc[groupname].to_dict()
    idf = IDF(os.path.join(root_dir, r"model_week4\model_week4.idf"))
    idf = modify_idf(idf, modify_dict)
    ...
```

### Key insights:

- **Folder creation:** Each run’s subfolder prevents overwriting.
- **Working directory:** Running inside each folder ensures outputs stay contained.
- **Cleanup:** `shutil.rmtree()` removes temporary files; comment it out while debugging.
- **Progress feedback:** `tqdm.trange()` gives a progress bar for batch runs.
- **Result collection:** Always store results in `result_df` before deleting folders.

 **Best practice:** Confirm outputs are saved successfully before cleanup.

## Troubleshooting

Issue	Likely Cause	Solution
IDD error	IDD not loaded	Call <code>IDF.setiddname()</code> with full path to <code>Energy+.idd</code>
No <code>.eso</code> file	EnergyPlus run failed	Check <code>eplusout.err</code> inside run folder
Unrealistic results	Wrong parameter mapping	Check <code>MAP_WWR</code> or <code>modify_idf</code>

## Deliverables

Save:

1. `week4_results.csv`
2. `week4_all_parameters.csv`

## Learning Progression

Part 1 and Part 2 together teach you how to:

- Automate EnergyPlus runs programmatically using Eppy.
- Manage file structure, weather data, and output logging.
- Generate multi-dimensional simulation datasets for analysis.

These skills form the foundation for **Part 3: data processing, visualization, and predictive modeling**.