



Geo-X training course

Hands-on 6

Please use the following citation for:

- **This exercise**

Miguel Sanchez-Lopez, Michaela Rychetska, 2026, Hands-on 6: Advanced Options – Geo-X Training Course (Version 1.0.). Climate Compatible Growth. DOI: [to be assigned].

- **Geo-X software**

Sanchez-Lopez, M., Naqvi, S., Leonard, A., Schirren, L., & Hirmer, S. (2026). Geo-X: Generalised geospatial software for production of green commodities. *SoftwareX*, 34, 102712. <https://doi.org/10.1016/j.softx.2026.102712>

Learning outcomes

By the end of this exercise, you will be able to:

- Check out the feature/flexible_demand branch and understand how it extends the main branch.
- Configure and run a generic generator to represent a power purchase agreement or existing low-carbon fleet.
- Configure flexible demand using the three h2_demand_flex parameters and interpret its effect on storage costs.
- Approximate salt cavern storage for a set of hexagons, and quantify the competitive advantage this creates.
- Compare advanced scenario results against your Hands-on 5 baseline.

This is the final hands-on of the course. Everything here builds on your ghana-h2-nh3 work from Hands-on 2 to 5, hydrogen only, using the full-country run from Hands-on 5 as your baseline.



Activity 1 – Check out the feature/flexible_demand branch

The advanced options in this exercise, the generic generator and flexible demand, live on a feature branch on CCG's upstream repository, not on main. You will check it out, then branch off it for your own work, the same way you branched off main back in Class 2.

Try it:

- Check whether your fork already has the CCG repository set as upstream: `git remote -v`. If you do not see an upstream entry, add it: `git remote add upstream https://github.com/ClimateCompatibleGrowth/Geo-X.git`
- Fetch and check out the feature branch: `git fetch upstream`, then `git checkout -b feature/flexible_demand upstream/feature/flexible_demand`
- Branch off it for your own work, do not edit `feature/flexible_demand` directly: `git checkout -b my-analysis/ghana-flex`

Note: this keeps CCG's feature branch clean, your config edits stay in your fork, traceable and comparable across scenarios, and any improvements you make can still be contributed back upstream through a pull request. Same habit as Class 2, one branch further down the tree.

Activity 2 – Configure and run a generic generator

The generic generator is a dispatchable electricity source added to every hexagon. Unlike solar and wind, it can generate at any hour, so the optimiser calls on it whenever renewable output is insufficient. Its capacity per hexagon (`p_nom`) is read from the hexagon GeoJSON; its price (`marginal_cost`) is set once, in `config.yaml`, for every hexagon. It can represent a PPA contract price, a local grid tariff, or the shadow cost of existing low-carbon generation.

Try it:

- In `config.yaml`, add `'existing'` to your `generators_dict` and set its capacity:

```
generators_dict: {'solar': [], 'wind': [], 'existing': []}
gen_capacity:
  solar: 1
  wind: 4
  existing: 1

generic_generator:
  marginal_cost: 60
```

- Run it: `snakemake -j 4 map_all`, then copy your results aside so the next run does not overwrite them: `cp results/hex_total_cost_GA_2023_hydrogen.geojson results/hex_total_cost_GA_2023_hydrogen_ppa60.geojson`
- Change `marginal_cost` to 100, rerun, and copy those results aside too, this time as `..._ppa100.geojson`.

Note: `p_nom` for the generic generator is drawn from hydro and nuclear capacity in your `hex_final_GA.geojson`. Ghana's hexagon file does not currently include those columns, so if you see no change at all between scenarios, that is expected, not a bug: it means the generic generator has no capacity to draw on anywhere in the country yet. Treat this activity as a methodological walkthrough of how the mechanism works, ready to apply once that data is added.

A low marginal cost (60) should let the generic generator displace a significant share of renewables and storage; a marginal cost closer to renewable LCOE (100) should mainly displace storage while renewables stay in the mix. A high marginal cost (140, not needed here) effectively switches the generic generator off and returns you to the baseline.

Activity 3 – Configure flexible demand

In the baseline, the hydrogen plant must produce at a constant hourly rate, which requires storage to smooth variable renewable output. With flexible demand, the plant can vary its output over a defined window, as long as it meets total demand across that period. Economically, this acts like a free storage buffer on the demand side.

Try it:

```
h2_demand_flex: True
h2_demand_flex_hours: 168
```



```
h2_demand_flex_power_fraction: 0.4  
h2_demand_flex_energy_fraction: 1.0
```

- Set these four lines in config.yaml (with `marginal_cost` and `existing` back out of the picture, so you are testing flexibility on its own).
- Run: `snakemake -j 4 map_all`, then copy your results aside as `..._flex.geojson`.
- Reuse the stacked bar code from Hands-on 5, Activity 5, on this new file. Compare the `storage_cost` share against your baseline.

Note: not every offtake is a good candidate for this. Ammonia synthesis via Haber-Bosch is inherently inflexible, while a port terminal with buffer tanks before ship loading, or an industrial process with holding capacity, can absorb timing variability on the buyer's side instead of the plant's.

Activity 4 – Approximate salt cavern storage (a theoretical exercise)

Salt caverns are underground cavities in thick salt deposits, increasingly considered for hydrogen storage at scale. Their availability depends on geology, not on renewable resource quality, so a location can be a poor solar or wind site and still be an excellent storage site. Ghana does not currently have a geological survey of eligible hexagons in this course's dataset, so this activity is explicitly a thought experiment: *imagine* salt caverns exist in two different parts of the country, and compare what that would do to the cost map.

- **Scenario A — South:** caverns near Accra itself, in the same region as demand.
- **Scenario B — North:** caverns far from Accra, in the north of the country.

Try it:

- In your notebook, split Ghana's hexagons into the two theoretical eligible sets by centroid latitude:

```
import geopandas as gpd  
  
hexes = gpd.read_file("data/GA/hex_final_GA.geojson")
```

```
hexes["lat"] = hexes.geometry.centroid.y

south_cavern = hexes[hexes["lat"] < 6.5]    # near Accra
north_cavern = hexes[hexes["lat"] > 9.5]    # north zone

south_cavern.to_file("data/GA/hex_final_GA_southcavern.geojson", driver="GeoJSON")
north_cavern.to_file("data/GA/hex_final_GA_northcavern.geojson", driver="GeoJSON")
```

- Back up your full hexagon file: `cp data/GA/hex_final_GA.geojson data/GA/hex_final_GA_full.geojson`
- Swap in the south subset: `cp data/GA/hex_final_GA_southcavern.geojson data/GA/hex_final_GA.geojson`
- In your `basic_hydrogen_plant` parameters, lower the CAPEX in `stores.csv` for hydrogen storage, to reflect the lower cost of a salt cavern relative to a compressed tank. Use a value your class agrees is realistic, and note it down alongside your results.
- Run: `snakemake -j 4 map_all`, then copy results aside as `..._southcavern.geojson`.
- Restore the full file, swap in the north subset instead, and repeat: run, then copy results aside as `..._northcavern.geojson`.
- Restore your full country file at the end: `cp data/GA/hex_final_GA_full.geojson data/GA/hex_final_GA.geojson`

Note: this is an illustrative approximation of the method described in Class 6, not a real geological assessment of Ghana. Treat the results as a demonstration of how cavern access could shift the cost map, not as a siting recommendation.

Activity 5 – Compare your scenarios

You should now have five result files beyond your Hands-on 5 baseline: `ppa60`, `ppa100`, `flex`, `southcavern`, and `northcavern`. Reuse the heatmap code from Hands-on 5, Activity 4, to plot each one, and compare them side by side.

```
import geopandas as gpd
import matplotlib.pyplot as plt

scenarios = ["baseline", "ppa60", "ppa100", "flex",
```

```

        "southcavern", "northcavern"]

fig, axes = plt.subplots(2, 3, figsize=(16, 10))
for ax, name in zip(axes.flatten(), scenarios):
    suffix = "" if name == "baseline" else f"_{name}"
    df = gpd.read_file(
        f"results/hex_total_cost_GA_2023_hydrogen{suffix}.geojson"
    )
    df.plot(column="total_cost", cmap="viridis", legend=True, ax=ax,
            missing_kwds={"color": "lightgrey"})
    ax.set_title(name)
    ax.axis("off")

plt.tight_layout()
plt.show()

```

- Which scenario gives Ghana the lowest cost at Accra? Is it the PPA, the flexible demand, or one of the two cavern scenarios?
- Does the north cavern scenario change the ranking of any hexagons far from Accra, even though transport cost is higher there?
- If you had to recommend one lever to a policymaker for reducing Ghana's hydrogen cost, which would you pick, and what would you want to verify before recommending it?

Wrap-up

You have completed this exercise, and the Geo-X training course. Before closing, check that:

- You are working on my-analysis/ghana-flex, branched off feature/flexible_demand.
- You have run and saved results for both PPA marginal costs (60 and 100).
- You have run and saved results for flexible demand, and compared its storage cost share to baseline.
- You have run and saved results for both theoretical salt cavern scenarios, and restored your full hex_final_GA.geojson afterwards.
- You can compare all scenarios against your Hands-on 5 baseline and explain what drives the differences.



Well done! Over six hands-on sessions you have gone from installing Geo-X to running and interpreting a full green hydrogen and ammonia case study for Ghana, and extending it with advanced generation, flexibility, and storage options. That is the complete Geo-X workflow, end to end.

If you would like to take this further, or have questions about the model, you can reach the Geo-X team directly:

- Alycia Leonard: alycia.leonard@eng.ox.ac.uk
- Stephanie Hirmer: stephanie.hirmer@eng.ox.ac.uk

Thank you for following the course.