Assignment 5: Complex capacity expansion planning

Tom Bearpark

November 2020

Replication, code, and folder structure notes

To replicate the .csv files and plots submitted with this assignment, please see the following notes...

1. There are six Julia scripts, and one R script[1] submitted with this assignment. All code is located in the `/code/` sub-folder in the submitted zip file.
2. The julia scripts include three run scripts: `5_code_q1.jl`, and `5_code_q2.jl`, and `5_code_q3_CES.jl`.
   - These scripts are used to run functions and save outputs.
   - Each of these scripts first loads functions that are saved in their respective functions file, so there are six `.jl` files submitted in total.
   - For example, `5_code_q1.jl` uses the functions in `5_functions_q1.jl`.
3. To run one of the master scripts, you will need to:
   (a) Ensure you have all of the required packages downloaded in your local Julia system. If not, you can donwload them using Julia's package manager.
   (b) Edit the two path strings in the top of each script you want to run...
      - The `pso_dir` variable should point to the location of the power systems optimization git repo on your machine. This path is needed in order to load the relevant input data provided by Professor Jenkins - and will be user specific.
      - The `wd` variable should point to the location on the users machine where this zip file is stored. This directory will be used to save output csv files and plots, and to load csv files used in post-processing (eg producing plots and tables).
   (c) You should not need to touch any of the functions files.
4. The **folder structure** in the zip file:
   - .csv files contained in the top level of `results/` subfolder are used in this write up for making tables presented in this document, and are outputs requested in the questions.
   - `results/data/` contains all csv files outputted by the solutions functions.
      - Within this folder, the structure of the results is:
        `results/data/{question number}/{number of hours used}/{carbon tax?}/`
   - `results/figs/` contains all figures used in this write up.

[1] I used an R script to produce some of the more complex plots using the ggplot2 R package.

Question 1

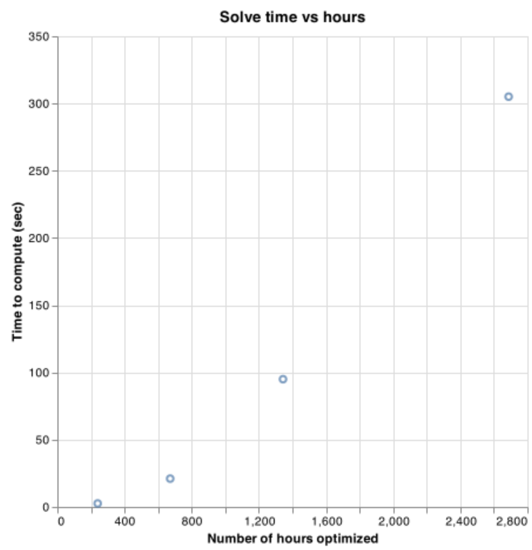
Part 1 (a)

To implement this model, I first created functions contained in `code/5_functions_q1.jl`, which are largely copied from Notebook 07. Functions in this file include a function for loading and formatting the input data, a function for running the model, and various helper functions for formatting and saving results. Please see inline comments in this file for more details. As a quality control, I compared the Total Cost value outputted by my function for running the model for 10 days worth of data to that in Notebook 07, and find them to be the same (as expected). To run code for question 1, please see `code/5_code_q1.jl`.

Part 1 (b)

Next, I run the model with different numbers of input hours. Output csvs for these model runs are saved in `results/data/question_1/{time_subset}_Thomas_Bearpark/without_carbon_tax/`.

The below plot scatters the time taking for the model to run against the number of hours included in the model...



We can see that solving the model for 10 days (240 hour) takes less than 4 seconds, whilst solving it for 16 weeks (2688 hours) takes over 300 seconds.

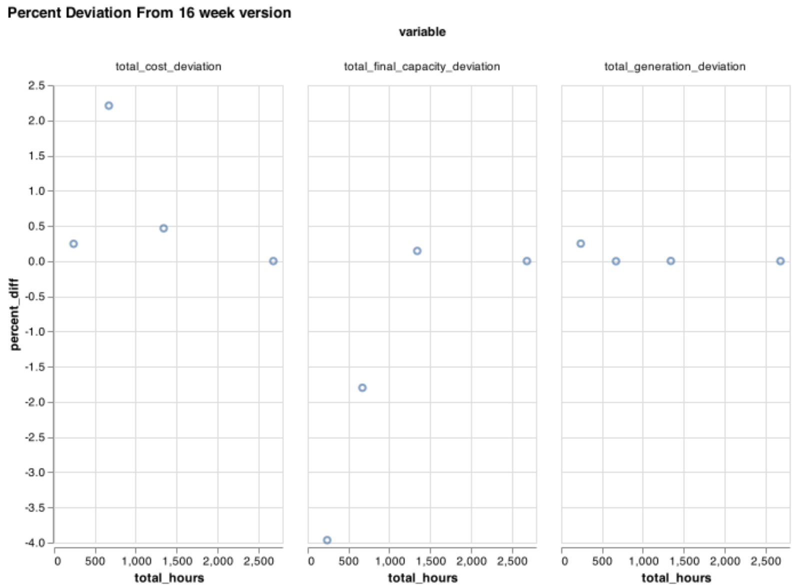
From the graph, we can see some evidence of non-linearity in the relationship between computational time and number of hours included. The increase in the time to compute associated with an additional hour included in the optimisation problem appears to be increasing in the number of hours. This non-linear relationship is to be expected (as discussed in the lectures), due to the curse of dimensionality.

Part 1 (c)

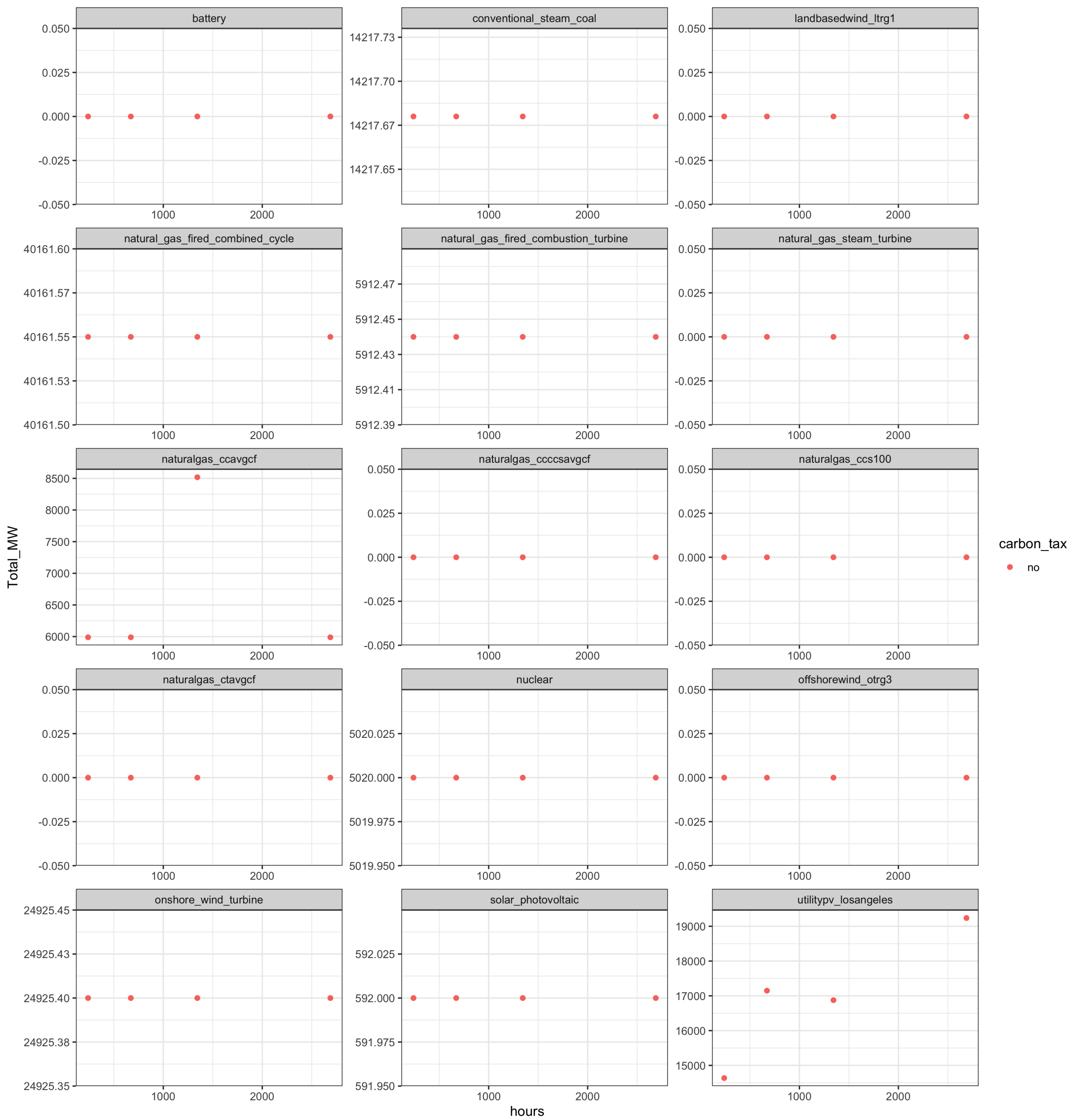
The below table compares the total cost results, total final capacity, and total generation in each version of the model runs described above.

Time Subset	Total Hours	Total Cost ($)	Total Final Capacity (MW)	Total Generation (GWh)
10 days	240	14197	111454	455891
4 weeks	672	14475	113966	454752
8 weeks	1344	14228	116223	454769
16 weeks	2688	14162	116055	454761

In order to visualise help us these results, the next plot shows the percentage deviation in each of these variables, compared to the value obtained from the 16 week optimisation. This is one way of assessing accuracy of the models - comparing them against model runs that use more data. Ideally, we would also compare them all against model runs with even more representative hours.



We can also break down how total generation by resource type varies across these model runs...



To see a spreadsheet of the values included in the above plot, please see:
`/results/q1_gen_by_resource_without_carbon_tax.csv`.

Interpretations

A number of features of the table and plots stand out...

- All model runs show less than a 4% difference, compared to the 16 week run, in the aggregate variables considered (such as Total Cost, Total Final Capacity, and Total Generation). Given the other uncertainties present in this analysis (i.e. uncertainties in the input parameters, policy uncertainty, etc), this accuracy loss may be relatively small.
  - There are some exceptions to this, such as for fixed transmission costs, and the installed capacity of utilitypv_losangeles and Naturalgas_ccavgcf.
  - However, these results do suggest that for the purposes of research questions that focus on the aggregate variables, using a time reduction method may be justified by the significant computational cost reduction.
- Using fewer representative hours seems to imply that the model overstates the total system cost.

- There is, however, a non-monotonicity in this relationship, in that the 4 week version has a higher total cost than either the 10 day version or the 8 week version.
- Looking at the breakdown of cost[2] for each run, it seems that the extra cost for the 4 week version is being driven by higher variable costs. The 4 week version also has the highest NSE cost of any of the models.
- Transmission fixed costs vary significantly across the model runs. This suggests that the locations of the demand and supply are important here. This might also account for some of the non-monotonicity shown in the first plot above.

- Runs with fewer representative hours seem to build less total capacity, in this model. The reasons for this can be seen in the plot of installed capacity by resource type above. Model runs with more representative hours build more 'utility_pvlosangeles'. Since this is renewable generation, it requires more higher installed capacity in order to reliably meet demand.
- Total generation is relatively constant across model runs.
- In general, the relative generation by each fuel type is maintained across the model runs. Almost all resource types are pretty much constant across model runs. The exceptions to this are losangeles solar, and natural gas.
  - Model runs with higher number of representative hours generally build more solar in this case. This seems to be a feature of the time sampling method, and could be related to the approach taken to time coupling constraints. Smaller representative hour subsets seem insufficient to capture the full value of solar.
  - Depending on which representative hours are picked, renewables may look more or less attractive to the system.
  - This suggests we should take probably take extra care over our time sampling methods in high renewable scenarios.

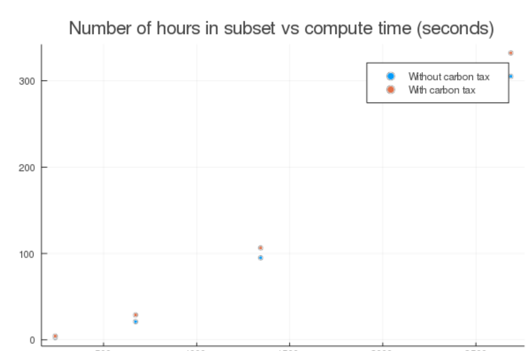
Part 1 (d)

To code up this question, I added an option to the `prepare_inputs()` function used in part 1(a) to allow for inclusion of a carbon tax. Please see line 146 of `/code/5_functions_q1.jl` for this implementation. I did this rather than copy and paste code as suggested in the question to try and limit the scope for my own error in manual copy and pasting and maintaining multiple versions of the same function during code development.

In this subsection, I first present some of the results obtained when running the model with the inclusion of a carbon tax, and then answer the specific questions asked in the assignment.

Part 1 (d) Results

The below plot shows how total solution time varies with the number of hours included in the model, for the model with a carbon tax, and for the model without a carbon tax.

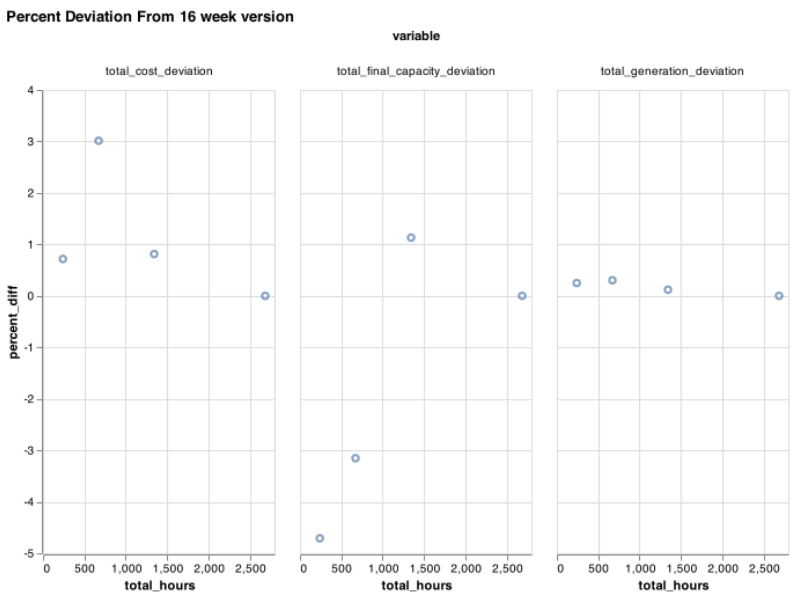


[2]Please see `/results/q1_cost_break_down_without_carbon_tax.csv` for details of this breakdown

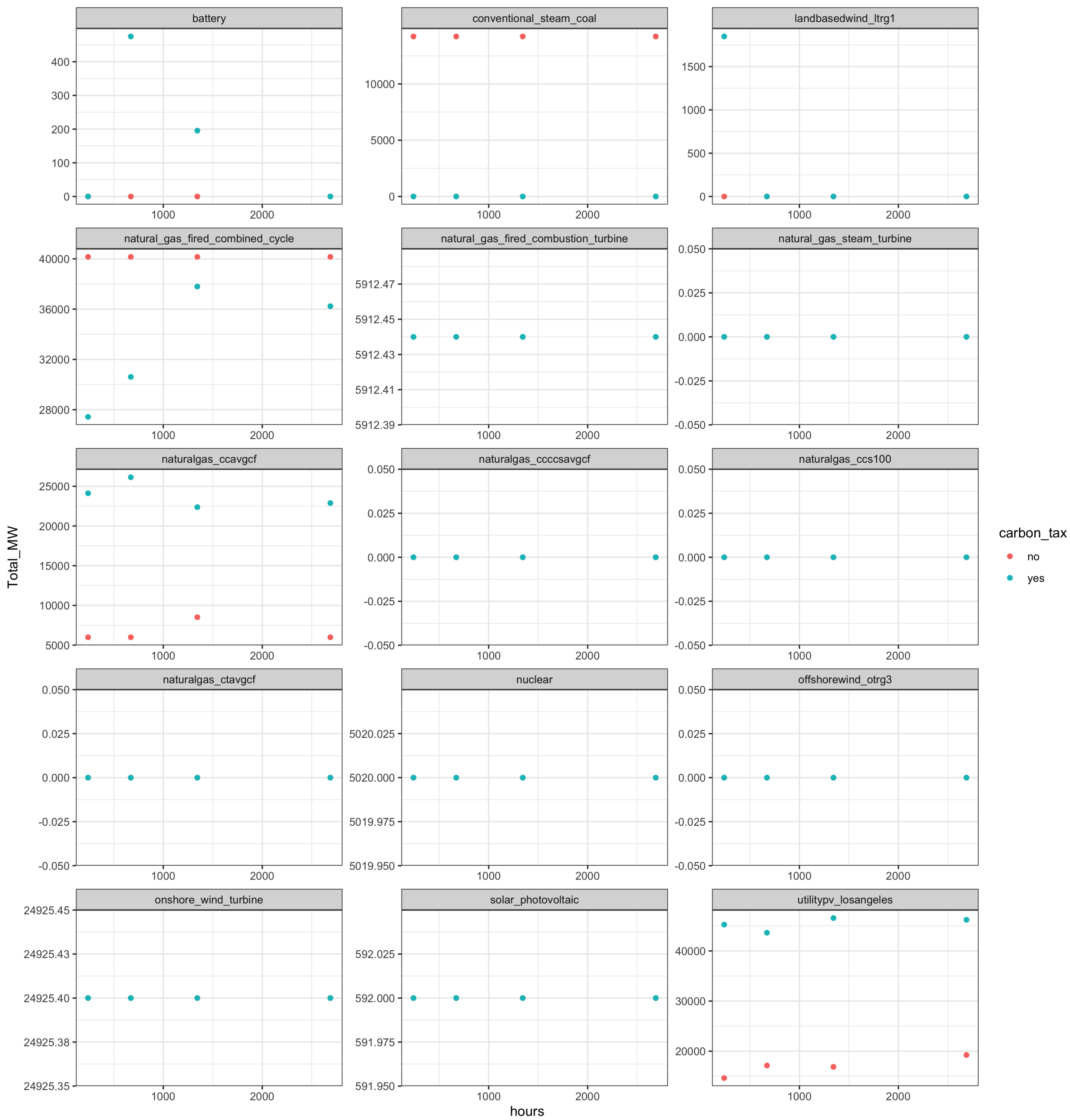
Importantly, the difference in compute time between the model with and without the carbon tax seems to increase as the number of hours included increases.

As before, we present the total cost, total final capacity, and total generation for each of the model runs, and a plot of their deviations from the 16 weeks model run (as a very rough proxy for model accuracy)...

Time Subset	Total Hours	Total Cost ($)	Total Final Capacity (MW)	Total Generation (GWh)
10_days	240	19280	135108	455915
4_weeks	672	19719	137314	456162
8_weeks	1344	19298	143389	455328
16_weeks	2688	19143	141786	454785



Finally, I present the generation by resource type results, overlaid with the results from without the carbon tax for comparison. For a spreadsheet of these numbers, please see:
`/results/q1_gen_by_resource_without_carbon_tax.csv`.



Part 1 (d) Answers to questions

How have the capacity and energy results changed overall with addition of the carbon price to fuel costs (relative to the original cases?

- Adding the carbon tax increases the amount of installed capacity significantly. Total final capacity increases from around 115000 MW to to 135000 MW. This trend is robust across sampling day subsets. The reason for this is that more renewable generation is installed, and this renewable generation is not firm.
- However, total energy generation does not change much with the addition of the carbon tax. This implies that we do not have a large amount of curtailed generation, or times when the system fails to meet demand.
- We can see from the breakdown of resources plot that the generation mix changes a lot when we implement the carbon tax.
  - We get a very large increase in the amount of installed utilitypv, and in natural gas. Conventional steam coal, on the other hand, is largely phased out of the generation mix.
  - There is also a significant uptick in the installed capacity of some natural gas types.

  - In some model runs, we also get battery storage installed. However, this is not robust across the different time samples.

How does the variation in the cost, capacity, and energy ouputs change now as you consider different number / duration of sample periods?

- The trends shown in the Percent Deviations plot above are similar with and without the carbon tax. However, the magnitudes of the percent deviations are larger.
- As discussed in part 1, it seems that renewable resources are more variable across the different samples. Given that the carbon tax has induced a more renewable heavy system, it makes sense that this system is more volatile across model runs with different number of sample days.
- In this scenario, with a carbon tax, we see very high variation in the installed capacity of some resources. For example the behaviour of battery installation is very inconsistent across model runs.
  - If we are interested in considering battery generation, with its time coupling constraints, that probably means we need to take extra care with sample selection methodology, and may present an argument for using more representative days in analysis of battery installation in highly renewable systems.

What does the overall experiment in Question 1 tell you about the generalizability of time sampling methods?

- We have seen that the accuracy of time sampling methods may not be constant across different policy scenarios. The inclusion of a carbon tax increases the magnitude of percent deviations from the 16 week version, probably due to the relative sensitivity of renewable generation build. The carbon tax implies that more renewewable generation is built, and time sampling is more sensitive under high-renewable scenarios.
  - This suggests we should be cautious about the generalizability of time sampling methods to other policy scenarios and settings, since we could once again find a differential effect.
  - Other dimensions through which accuracy is affected through time sampling could exist, for example, if we are considering a system with more storage. Our time sampling results may not be generalizable to this setting, due to the additional complexity involved, and required research simplifications in dealing with time coupling constraints.
- Throughout the problem, it became clear that there were some seemingly anomalous results from the 4 week sample. This suggests we should be cautious, as there can be some non-monotonicity in cost trajectories of results as we increase the number of sampling days included.
- I also found that compute time is sensitive to inclusion of a carbon price. This suggests that other policy scenarios and model environments could affect computation time (non-linearly), and therefore that compute time might not be extrapolated well across different modelling situations and policy scenarios.
- On the other hand, time sampling methods do seem to perform well for aggregate variables, and in regards to (most) thermal generation capacity. Also, they reduce total compute time non-linearly - which is an important consideration for a resource constrained researcher.
  - As usual, the conclusion to whether the time sampling methods are generalizable is **it depends on what research question we are interested in**. A careful evaluation of whether the policy question being researched is likely to be affected by a smaller sample of days is certainly warranted.

Question 2

Part 2 (a)

To run code for question 2, please see `code/5_code_q2.jl`.

This code uses functions contained in `code/5_functions_q2.jl`. Please see inline comments in this file, where I reference the specific instructions followed in implementing each part of the question, for details of how I implemented the instructions in part (a). The changes are in the `solve_model_q2()` function, which is in `code/5_functions_q2.jl`, from line 246 onwards.

For details changes made for each instruction...

0. See line 236. Note, I also included a tolerance of 1%, as in the class notebooks.
1. See line 254-255
2. See line 267-272
3. See line 350
4. See line 354
5. See line 356
6. See line 372-378
7. See line 417-420
   - This also required a wrap around constraint. Implemented in lines 422-423.
8. See line 430-441
9. See line 455-480.
   - This also required a wrap around constraint. Implemented in lines
10. See line 485-493[3]
11. See line 383-391
12. See line 500-502 for expression for start costs. This expression is then included in the objective function in line 533-536.

Note - I also updated the definition of the *eFixedCostsGeneration* expression, to include the correct definition of UC generator investment. See line 492 for details.

Part 2 (b)

To answer this question, I first present plots and tables of the results, and then provide interpretations that answer the questions presented in Question 2 (b).

The below table shows how fixed costs (abbreviated as F.C in the table) and variable costs vary for the model run with 8 weeks of data, compared to the model run in part 1 (both models do not include a carbon tax).

Model	Total Costs	F.C.Gen	F.C. Storage	F.C. Trans	Variable Costs	NSE Costs	Start costs
Without UC	14227.9	5186.5	0.0	132.4	8884.4	24.6	NA
With UC	14249.5	5115.6	0.6	136.2	8967.7	27.7	1.8

We can also look at how the total installed capacity and generation vary across the two models.

[3] I think that this constraint should include a wrap around version, since it is a time coupling constraint. However, unfortunately I did not have the time / could not work out how to code this. Instead, I defined these constraints over the full set of hours that are included in each model run. This will bring some level of (I think relatively small) abstraction error. Time periods near the starting edge of the time set should have larger minimum bounds on these constraints than are implemented in my version.

Model	Total Cost	Total Capacity	Total Generation
Without UC	14227.9	116222.9	454768.9
With UC	14249.5	115517.2	454804.6

Finally, the below plot shows a breakdown of the differences in generator types by resource used in the two optimal solution of the two systems.

Interpretation

- In general the solution does not change much, in terms of the aggregate variables, due to the addition of UC constraints.
  - Total costs are very similar, as are total capacity and generation.
  - There are slightly more NSE Costs in the model without UC than with UC. This may reflect the differences in optimal installed capacity.
- However, there are some important differences for certain features of the model.

- For example, without UC constraints, no battery storage is built, but with them, we build 40MW. This is significant, if this model was being used to analyse potential battery subsidy policies, for example. This change may be due to the system with UC needing the extra flexibility, given the UC constraints reduce system flexibility somewhat.
- Another key difference is in the development of *naturalgas_ccavgcf* capacity. This is signficantly higher with UC constraints, seemingly displaying *los angeles utility pv* capacity. This again may reflect the increased value of building flexible capacity in a world with UC constraints, relative to the intermittant renewable generation.

- Whether we should include UC constraints will depend on the research question we are analysing.
  - We should note that including the integer UC constraint increases run time hugely. This model took more than an 80 minutes to run on my laptop.
  - It might be appropriate to ignore UC constraints when looking at research questions that focus on overall aggregate system cost and capacity. Also, given how close the solutions are here, a research question that doesn't need a particularly detailed answer, with a detailed capacity breakdown, may be justified in not considering UC constraints. This might be the case in many policy scenarios.
  - It may be less appropriate when considering research questions related to energy storage capacity, or to specific subsidies for renewables and the natural gas plants that they are competing with.
  - Note - these conclusions are tentative, more research and literature review would be required to decide if this is robust to other modelling contexts, or if these conclusions are just a feature of this particular modelling setting.

Part 2 (c)

Next, we can implement a linear relaxation to the model ran above, where we no longer force any variables to be integers. To implement this, I added an option to the `solve_model_q2()` function, to allow for a linear solution. See line 274 of `code/5_functions_q2.jl` for details of this implementation.

The below table shows the solution time for the three models considered in this question. "Without UC" refers to the model ran in Question 1, which doesn't include Unit Commitment constraints. "With UC" refers to the model ran in Question 2(b) which includes integer decision variables, and UC constraints. "Linear Relaxation" refers to the model with UC constraints, but without integer decision variables.

Model	Solution Time (s)
Without UC	95
With UC	4823
Linear Relaxation	256

This shows there is a huge difference in compute time brought about my turning this into a MILP, rather than a linear problem.

The next table shows the costs results for the UC model with the integer variables vs the linear relaxation, and the model ran without UC in Question 1...

Model	Total Costs	F.C.Gen	F.C. Storage	F.C. Trans	Variable Costs	NSE Costs	Start costs
Without UC	14227.9	5186.5	0.0	132.4	8884.4	24.6	NA
With UC	14249.5	5115.6	0.6	136.2	8967.7	27.7	1.8
Linear Relaxation	14247.6	5091.7	0.0	135.0	8986.8	32.3	1.7

Finally, I present the generation capacity installed by resource type for each of these three models...

Interpretations

- The solution for the linear relaxation version with UC is very similar to the solution found in question 1 without UC.
  - The generation capacity build for the linear relaxation is qualititavely very similar. As are most other dimensions I looked at.
- The linear relaxation, like the UC integer problem, has some start costs that are not considered in the version from question 1. These are estimated to be quite similar in the linear relaxation case to the full integer UC model.
- It seems, however, that a linear relaxation is not sufficient to capture the effects of UC constraints on optimal storage build in the presence of UC, or on the optimal solar / gas capacity. These are the major areas of difference from the integer UC problem.
  - Intuitively, this might be because the lumpy nature of the integer constraints reduces the flexibility of the system, in such a way that affects the optimal capacity of storage and renewables.

  - If our research proposal requires a detailed breakdown of renewable and storage optimal capacity, it seems that we might want to include the UC integer constraints.

- The computational cost here from including UC constraints is highly significant. This means that even if there is some accuracy loss from abstraction, a researcher may have to make this trade off if their project requires many model runs (e.g. if they are doing MGA).

- Once again, further research / literature review into the generalisability of these conclusions to other modelling environments are warranted.

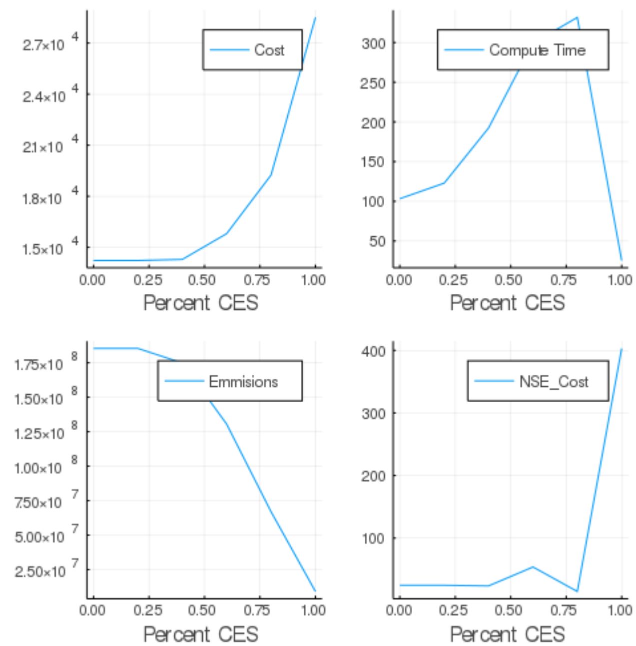
Question 3

To run code for question 3, please see `code/5_code_q3_CES.jl`. To implement this question, I added an option for CES stringency to the function `code/solve_model_q3()` in the file `code/5_code_functions_CES.jl`.

The constraint added to the model is of the form:

$$\sum_t \sum_{g \in CES} Gen_{t,g} \geq \gamma \sum_t \sum_{g \in G} Gen_{t,g}$$

Where CES denotes the set of generators eligible for the Clean Electricity Standard, and $\gamma \in \{0, 0.2, 0.4, 0.6, 0.8, 1\}$ is the stringency of the proposal. This policy constraint requires that the total proportion of CES generation used in each model run is equal to γ. The below plots show how some key outputs vary with γ.



Total system cost increases non-linearly with γ. This is intuitive, and corresponds to the results presented by Professor Jenkins[4] in class. At high levels of renewable penetration, the cost of an extra unit of emmisions reduction is higher, due to system value of a diverse capacity portfolio. On certain days of the year, when renewables are not efficient to use as generation (i.e. when the weather is dark and not-windy), a very high renewable standard could be very costly.

We can see that **compute time** is increased by increased stringency, except for when we have $\gamma = 1$. When the constraint requires that CES makes up 100% of generation, this actually simplifies the problem, since we have fewer possible generation technologies available, and the solution becomes much faster. Up until this point, the solution is becoming more complex, hence the higher solution time.

NSE costs are relatively flat, and then jump to a high level at a 100% CES. This reflects the difficulty in meeting demand at very high renewable penetration.

Emissions are not affected by the CES policy at first, since it doesn't bind at low levels of γ. Then, emmisions fall linearly with γ, down to a level nearly 10 times smaller than in the base case. Note, emissions are not zero at $\gamma = 1$, since qualifying for the CES doesn't mean that a fuel is exactly zero emissions.

[4] `https://www.cell.com/joule/pdf/S2542-4351(18)30386-6.pdf`