2022-01-13

Archive folder has some of the previous model iterations.

Results folder has some of the output and plots from the model output.

Otherwise, everything is currently in the main “hydrogen” directory.

Also note that the references, calculations for HDV demand, existing infrastructure, cost and efficiency data for equipment, etc., is located in the Box folder.

\*\*\*

Here are some of the important things in the Box folder:

Demand Segments/

truck\_fueling\_stations\_map.svg shows the data used for estimating HDV demand

truck\_traffic\_estimate.xlsx shows the calculations of the HDV demand

Existing H2 Supply and Infrastructure/

H2 Supply/

Has some data we got from shell. They scrubbed some proprietary stuff to give us an approximate version of the existing hydrogen production capacity in Texas.

Model/Methods/

PH\_Model\_Methods\_v##.doc gives an overview of the equations being used in the model. V14 is a bit dated, but mostly reflects the equations being used in the model. The one thing it’s definitely missing is the infrastructure subsidy code, but that’s somewhat straightforward depending on how we model that going forward. Thomas might take a crack at updating these methods in early 2022 while it’s fresh in my mind.

Data\_brainstorm/

Has a lot of the calculations used to parameterize the various production, conversion, distribution technologies in the model.

Model Backup/

2022-01-13 hydrogen\_v33.zip is a backup of the most recent version

Report/

2021-09 Texas Hydrogen Roadmap Draft v\_21.docx is the most recent version of the report outline. It should be updated based on the 2022-01-13 DOE presentation.

Pathway visuals/

Contains the svg and png files of some of the pathway art I developed

\*\*\*

Here are some important things in the hydrogen model folder

There are three important .py files to be aware of:

**create\_graph.py**

This generates the networkx object that holds all of the data about the different nodes and the arcs that connect them. You can look around in there to see how the network is created.

This script runs automatically when you run the main hydrogen model script.

Basically it follows these steps:

1. Reads in the hydrogen data, which is passed to it as an object from the main hydrogen script
2. create\_graph: using the nodes, arcs, and distributors csvs, it creates a basic network that creates the hub nodes and connects them to each other. This creates a basic skeleton on which other technologies will be added later on.
3. add\_consumers: uses the node and demand csvs to create “demand\_sectors”—e.g., heavy duty trucks in Austin. It creates a node for that sector containing the data about its breakeven price, carbon sensitivity, size, etc., and then connects that demand\_sector node to the appropriate “demand” node, depending on the type of demand (e.g., low\_purity, high\_purity, fuel\_station).
4. add\_producers: creates nodes and arcs for existing and potential hydrogen producers, and connects them to the location’s low-purity or high-purity hub.
5. add\_converters: this one is a bit more complicated, but is meant to give flexibility to how we construct the other technologies within the network. It uses the node and converters csv. The converters csv is particularly important here, because the model reads through that csv row-by-row, sequentially adding technologies as it goes. This is important because some technologies cannot be added before something upstream (e.g., a liquid hydrogen terminal must be created after the liquefaction facility because it is downstream of it. The way it works is the model looks at the “arc\_start” and “arc\_end” columns of the conversion csv, and that tells the model where to put the new technology.

For example, with liquefaction, the model creates a liquefaction node, inserts it between the “hub\_highPurity” node and the “dist\_truck\_liquefied” node and adds new arcs. I.e., before the liquefaction node is created, the hub\_highPurity and dist\_truck\_liquefied nodes are directly connected with an arc. After the conversion script runs the “liquefaction” row of the conversion csv and inserts the liquefaction facility, we now have three nodes connected by two arcs: hub\_highPurity—liquefaction--dist\_truck\_liquefied. (Becuase we need to liquefy hydrogen and then send it through a terminal before it makes sense for it to enter the dist\_truck\_liquefied node so it can be sent on liquid trucks to local demand or other locations.)

Then, to add the terminal, row 4 of the conversion csv has the terminalLiquid technology with arc\_start=converter\_liquefaction (i.e., the node we just created in the last step) and arc\_end=dist\_truckLiquefied. So, it inserts a terminal after the liquefaction facility and before the trucking distribution hub.

In this way, you can add rows of conversion technologies to the csv and the model will build them according to the arc\_start and arc\_end values.

1. Add price nodes: this uses some of the hard-coded inputs from the main hydrogen script to create a series of dummy consumers with various breakeven prices. For example, in Austin, it might create 50 consumers with breakeven prices of 3.50, 3.55, 3.60. … 9.90, 10.00 with a demand of 1 kg. Then when the model solves, it will provide hydrogen to many of these dummy consumers, but not all. If it does supply hydrogen to the consumer with breakeven price of 7.65, but it does not provide hydrogen to the consumer with breakeven price of 7.60, then we assume the price for hydrogen in Austin is 7.65.

**discreteize\_deamnd.py**

this is an older script I used to basically create gaussian curves of consumers with different sizes and breakeven prices. It made demand more stochastic. But I found as we presented those results to stakeholders that the stochastic demand idea was more confusing than helpful.

I’m leaving the script in here if it’s useful in the future, but it’s not being used in v33 of the hydrogen model.

You can look in the archived models (maybe v20?) to see how it was used before.

**hydrogen\_v##.py**

the main hydrogen script. Currently v\_33, and v\_33fuelstationsubsidy which shows a particular scenario used for the DOE milestones results.

Preamble: gives comments on the evolution of the different versions. V30 had a lot of changes, taking away storage and discretize\_demand. I felt that those were making the model harder to interpret.

class hydrogen\_inputs:

holds a lot of the data directories and hard coded data that feed into the model. Note in particular.

Just after the class is defined, you’ll see it called:

H = hydrogen\_inputs(…then a bunch of arguments. The important arguments are:

Csv\_prefix: this should be “texas\_” if running the model and blank, i.e., “” if troubleshooting or adding features and you want to work with a smaller 2-node model.

Price\_tracking\_array: the array of prices that will be used in the “add price nodes” function of the create\_graph.py file. It’s good to set a range of values where you think the actual price will fall between.

Price\_hubs: only the nodes in this list will be added in the “add price nodes” function of the create\_graph.py file. If we try to find prices at every hub, it can make the model run slower.

Subsidy\_dollar\_billion: sets a cap on the amount of subsidies

Subsidy\_cost\_share\_fraction: the percentage of capital cost that industry must pay for

Then the model uses H to create the networkx object, g

Then it feeds g and H into the pyomo format using a concrete model

After that it defines sets. Many of these sets are based on how the network is constructed and on the class names of some of the different nodes. These sets are helpful for defining constraints, objectives, etc. that only apply to specific types of nodes or arcs in the model.

Then it creates parameters and variables.

Then it feeds those into the objective function.

Then it does all of the constraints.

Then it solves the model.

The code is pretty clean up to that point, and then there are a bunch of snippets of code on how to process the pyomo output and get some results from it. In particular the objects beginning with “r\_” (i.e., ‘r’ for ‘results’) have some good data. E.g., r\_prices will tell you the price at the different hubs, r\_liquefiers will tell you which hubs build liquefaction facilities and how big they are, etc.

You will probably want to clean the reporting up and push the output to a csv or an arcgis map or something, but there should be enough bones in there for you to work from.

\*\*\*

Running the model.

As it is, you probably need to comment out a lot of the reporting part of the script (the stuff after the model is solved. Once you do that, then you could run the script as usual as long as all the csvs and the create\_graph.py file are in the same directory. I usually run this from the IDE spyder and will do bits of code at a time, which is why it’s not in shape to run the whole script at this point from an ipython console.

\*\*\*

CSV Files

arcs connections between the nodes for a 2-node practice network

ccs data on ccs costs and effectiveness. Whether or not a production facility can use ccs is defined in production.csv

conversion cost, efficiency, and data on where to place the node for conversion technologies. Note that the technologies have a utilization value. If utilization is 0.90, then the model has to build a 100-ton facility to convert 90-tons/day of hydrogen.

demand generic information about different demand sectors

distribution generic information about different distribution technologies. Note that “unit” as in capital\_usdPerUnit is equal to one pipeline or one truck. So, if the model builds 20 liquid trucks to take hydrogen from Houston Austin, that arc in the model will say that the capacity is 20 but the flow is 160 (20 trucks X 8 tons daily capacity). Or, if it builds 2 pipelines, the capacity will be 2 and the flow (2 X 999999).

nodes nodes for a 2-node practice network. Contains information about whether a node can build a different production technology, and which types of demand sector exist at that node

production generic cost and efficiency information for different hydrogen production sources

production lets you define unique existing production technologies at existing nodes with unique costs and efficiencies

storage this is no longer being used in the model, but it was holding storage technology information

texas\_... nodes, arcs, and production\_existing csv files that are specific to the Port Houston model